Modeling Emotions with EEG-data in StateCraft

THESIS

By Anders Njøs Slinde

Department of Information Science and Media Studies

University of Bergen

Spring 2012

Abstract

Emotions have been shown to play an important part in human decision making, and emotions in Artificial Intelligence have been shown to affect agent performance and believability. The aim of this thesis is to use EEG-data to model players' emotions. The emotion model was incorporated into the existing Emotion module in the computer game known as StateCraft. Using artificial neural networks as a tool, two different models of the players' emotions were created, a general model and a country specific model, resulting in four different configurations of the Emotion module. Simulations of these four different configurations of the Emotion module were conducted.

Statistical analysis of the simulation data shows that the agents perform worse overall with emotions than without. The country specific model appears to perform better than the general model in the simulations. Analysis also indicates that the four new EEG-based configurations perform worse overall than the existing Emotion module which is based on game states. The EEG-based emotions promote more risky behavior, and for some countries that can have a very negative effect on performance.

Acknowledgements

I would like to thank my supervisor Weiqin Chen for her support and guidance.

Table of Contents

List of Tables

Chapter 1

1 Introduction

1.1 Introduction

The field of artificial intelligence (AI) in games has been gaining serious traction in parallel with the video game industry growing substantially over the last few decades. Researchers and game industry professionals have been working towards the common goal of creating autonomous intelligent agents that can perform as replacements for human players. The agents need to make good decisions, or realistic bad decisions, in order to appear like a human player. Emotions have been shown to have an effect on human decision making. Whether emotions can bring something of value to an autonomous agent's performance and decision making process is worth investigating.

In this thesis the board game Diplomacy will act as a platform to investigate the research questions. In 2006 Helgesen and Krzywinski implemented a computer version of Diplomacy, named StateCraft (Helgesen & Krzywinski, 2006). Students from the University of Bergen have continued to work on the StateCraft game. The latest project was done by Carlson and Hellevang in 2010 and resulted in an Emotion module and a Prisoner's dilemma module (Carlson & Hellevang, 2010). The Emotion module was designed based on data gathered from four interviews after one game of the board game version of Diplomacy. In their evaluation of the Emotion module it was found that some countries perform worse with emotions. The most obvious short coming of the Emotion module is that the emotion model was based on data gathered from interviewing a small set of people about only one game of playtime.

The Emotiv Epoc headset is used in order to combat this short coming. It provides Electroencephalography(EEG) data, which interprets into emotions and facial expressions (Emotiv, 2011). The StateCraft game is extended with an EmotivLogger module which takes the emotion data from the Emotiv Epoc headset and couples it with game data from StateCraft. This gives the opportunity to use "real" data about players' emotions when crafting the Emotion module. StateCraft was extended with capability to read the emotional state of the player and log it to files together with corresponding game states. These files were used as input to train an artificial neural network which models game state to emotional state.

The last part of the project involved using the model of player emotions and creating a new Emotion module. Because of the great ground work laid down by Carlson and Hellevang (2010) and the existing Emotion module was built upon and expanded. This made it possible to create different configurations of the Emotion Module which includes or excludes the emotions from Carlson and Hellevang's work in addition to the Emotiv Epoc emotions. Modeling emotions into the agent AI has huge potential to change the agent's decision making and the player's experience. Diplomacy's game play is a very social experience, and human social interaction is driven by emotion. Player's emotion in games has been largely ignored by the game industry and the research community. Only recently player modeling in computer games has begun to attract an interest from the research community. This makes the project interesting from both an industry stand point and a research stand point.

1.2 Research Questions

To evaluate the new Emotion module and its effects two research questions are investigated:
RQ1: How does modeling emotions of players affect agent performance?
RQ1.a: Does an agent perform better with emotions than without emotions?
RQ1.b: Does an agent using country-specific emotions perform better than an agent using general emotions?
RQ2: Does an agent trained from EEG-data perform better than an agent based on game states?
Using artificial neural networks as a tool, two different models of the players' emotions were created, a general model and a country specific model, resulting in four different configurations of the Emotion module. Simulations of the four different configurations of the Emotion module were run in order to answer these two research questions. The StateCraft autonomous player agents were put trough game simulations in order to evaluate the new Emotion module configurations. Four different new set ups of the Emotion module were evaluated.

1.3 Research Method

In order to answer the research questions this project was executed as a design research project, with iterations of development and testing. Design research design research was found to be the most fitting alternative for the project, since the project explores a very fresh and new field of research. There were no previous artifacts which implements what this project seeks to research, so creating the artifact before conducting the research was the only natural option. The project also uses a brand new technology, and this project aims to highlight some of the potential applications for the technology used. Hevner et al (2004) argues that artifact instantiation demonstrates feasibility both of the design process and of the designed product (Hevner, March, Park, & Ram, 2004). This project demonstrates one way EEG-reading devices can be useful in researching emotion in artificial intelligence.

According to Hevner et al (2004) a mathematical basis for designs allows many types of quantitative evaluations of an IT artifact, including optimization proofs, analytical simulation and quantitative comparisons with alternative designs (Hevner, March, Park, & Ram, 2004). In order to validate and test the theoretical design idea outlined here there is a need for an artifact. The research questions could only be answered by analyzing an artifact. In the evaluation phase of this project mathematical and statistical methods were used to analyze the different variants of the Emotion module this project produced.

The artifact developed also has potential to open up for new research opportunities in the future. By giving future researchers, and possibly University of Bergen students, an artifact to further study is a good motivation for conducting the research presented in this thesis. This enforces the belief that developing a good artifact has great research value.

1.4 Organization of the thesis

The thesis is organized as follows: Chapter two presents the theoretical basis for artificial intelligence and emotions used in the development and evaluation of the project. Chapter three introduces the board game Diplomacy and StateCraft, with a focus on the previous work on the Emotion module. Chapter four presents design and development of the three components of this project, the Emotion Logger, the Emotion Learner and the new Emotion module with different configurations. Chapter five contains an evaluation the new Emotion Module. In chapter six the conclusion of the thesis is made, as well as some suggestions for future work.

Chapter 2

2 Literature Review

2.1 Artificial Intelligence

We as humans have an idea of ourselves as the most intelligent life on earth. But what makes us intelligent, and how one would define intelligence is something we have not agreed on and maybe never will. John McCarthy, who first coined the term Artificial Intelligence, defines intelligence as "[..] the computational part of the ability to achieve goals in the world" (McCarthy, 2007). There exists such a thing as degrees of intelligence. If a machine is designed to perform a very well understood and formalized task it can give a very impressive performance on the specific task. Alan Turing is credited as being one of the first artificial intelligence researchers, giving a lecture on artificial intelligence in 1947 (McCarthy, 2007). There are countless of myths and theories of what artificial intelligence is and what it could be, a lot of these from movies and books in popular culture. We all know and love the two droids from the Star Wars universe, C3PO and R2-D, who have conversations, emotions ,relationships, and their own opinions on the world they inhabit.

2.1.1 Artificial Agents

C3PO and R2-D2 would be referred to as artificial agents by artificial intelligence experts. In the field of artificial intelligence one defines conscious, cognitive entities that have feelings, perceptions and emotions just like humans as artificial agents. More broadly one can describe an agent as anything that can perceive its environment through sensors and act upon the environment with actuators (Russel & Norvig, 2003). Artificial agents are automated and behave as they are designed and programmed to behave. One way to design and program an artificial agent implement a simple reflex-agents that have condition-action rules, for example "if warm then take jacket off". Model-based agents hold a model of what state the world is in and how the world changes independently of the agent. Even more advanced than this would be the goal-based agent which holds goal of how the ideal world should be, and also (possible partial) information on how its actions will change the world. This makes it possible for goal-based agents to reach a goal. In most environments goals alone are not enough to generate high-quality behavior (Russel & Norvig, 2003). Goals just create a binary value which describes if a thing was good or if it was not good. Utility-agents are therefore used. These agents have a utility function which takes a state and calculates a number that describes how "happy" the agent would be in that state. This makes it possible for the agent to know how to prioritize between objectives (Russel & Norvig, 2003).

2.1.2 Four approaches to Artificial Intelligence

Because of the controversy in the field of artificial intelligence has been split up into four different approaches. The four approaches are as follows:

- Artificial agents that **act like humans:** These systems are designed to pass the Turing Test, which was proposed by Alan Turing in 1950. In order to pass the Turing Test a human interrogator must be unable to distinguish the system from a human being . This means that the system needs the following AI systems:
    - Natural language processing in order to communicate with the interrogator.
    - Knowledge representation to store knowledge.
    - Automated reasoning to use the stored knowledge to answer questions
    - Machine learning to detect patterns and adapt to new information

  Turing deliberately avoided any direct physical interaction between the interrogator and the AI system.
- Artificial agents that **think like humans:** In order to make systems that think like us, we need to have some kind of model of how we, as humans, think . This field get inspiration from, and even works closely with, the field of cognitive science. Cognitive scientists have been able to create partial models on the workings of the human brain. Some of these models are discussed in a later chapter.
- Artificial agents that **act rationally:** The systems that act rationally always try to get to the best outcome from a given situation or, when best is not possible, the best expected outcome.
- Artificial agents that **think rationally:** This approach is also called "the laws of thought" approach. The laws of thought are the patterns for argument structures that always yield correct conclusions when given correct premises (Russel & Norvig, 2003).

2.1.3 Machine Learning

The field of machine learning is concerned with how to construct computer programs that can learn and improve with experience. Today machine learning is used for a wide range of applications such as; computer vision, natural language processing, search engine, medical diagnosis, computational finance, classifying DNA sequences, and more.
There is a set of branches of machine learning algorithms depending on the environment your agent is going to occupy.

- **Supervised learning:** These algorithms analyze the training data in order to create a target function that can predict the correct output value from any valid input value. The training data contains example pairs of desired output and input. These examples are representations of the environment the agent will operate in after the learning is complete. The training is often done offline[1] in supervised learning algorithms.
- **Unsupervised learning:** is often used to find hidden structures in unlabeled data. This means that the algorithms need to operate without an error or reward signal to evaluate potential solutions.
- **Semi-supervised learning:** uses both labeled and unlabeled training sets to generate the correct function.
- **Reinforcement learning:** tries to maximize the reward given. The environment serves the algorithm with states which the agent can act on. To guide the learning algorithm the environment gives out rewards which the algorithm uses to figure out which actions are the best to take given a certain environment.

2.1.3.1 Choosing the right Machine Learning Algorithm

In order to select the correct machine learning algorithm one must consider the attributes of the environment and the desired performance of the agent.

- **Training data:** The structure of the training data is very important to the machine learning algorithms. A key attribute of the training data is whether the training data

[1] Offline training means that the agent does not change the learned function once the initial training phase has been completed.

provides direct feedback regarding the action performed by the agent (Mitchell, 1997). A second important attribute is how the learning agent can control the sequence of the training data (Mitchell, 1997) . The training data may be provided by a process outside of the learning agents control, the learning agent may be able to query for specific scenarios in the training data, or the agent may be able to explore its own environment for training data (Mitchell, 1997). The third important attribute one must consider if the accuracy and relevancy of the training data. How well the training data reflects the environment the learning agent has to perform in.

- **Target function:** Choosing how to design the target function depends on how the function will be used by the agent, and what type of knowledge one wants the agent to learn. For example an agent wants to learn how to choose the best move in chess given a certain board. Then the target function will be a mapping from board to move (B -> M). The target function need to fit in with the behavior you want your agent to take, as well as fit with the training data you have available. Often one expects nothing more than an approximation from the target function (Mitchell, 1997).
- **Target function representation:** In choosing the target function representation the designer of the machine learning system or agent needs to prioritize its expressiveness. High expressiveness of the target function means that it will be a very close approximation of the ideal target function; on the downside this means that the training data needs to be more extensive (Mitchell, 1997).
- **Learning algorithm:** After deciding on a target function for the given training data, and a representation for the chosen target function a learning algorithm can be deployed in order to improve the target function. The learning algorithm one chooses to use depends heavily on the target function and the training data.

The following sections present machine learning techniques that were considered for this project. Based on prior field knowledge, some techniques were already excluded from being used, so they are not discussed here.

2.1.3.1.1 Bayesian Networks

Bayesian networks are networks of conditional probabilities. The name comes from Bayes who described a theorem for calculating conditional probabilities.

$$P(H_i|E) = \frac{P(E|H_i) \times P(H_i)}{\sum_{k=1}^{n} P(E|H_k) \times P(H_k)}$$

$P(H_i|E)$ is the probability that hypothesis H_i is true given evidence E. $P(H_i)$ is the overall probability of the hypothesis H_i is true. $P(E|H_i)$ is the probability of observing evidence E when H_i is true.

In order to deploy a Bayesian reasoning a few points needs to be fulfilled:

- All the probabilities on the relationship between evidence and the various hypotheses must be known.
- The probabilistic relationships among the pieces of evidence must be known (conditional independence of evidence)
- Relationships between evidence and hypotheses $P(E|H_k)$ must be calculated
- Rebuild probability tables when new relationships between hypothesis and evidence are discovered.

Bayesian methods can be used to determine which hypothesis is most likely given the set of evidence (Mitchell, 1997). The hypothesis found would be the most optimal in the meaning that no other hypothesis is more likely.

2.1.3.1.2 Reinforcement Learning

As described above an agent that learns through reinforcement learning will receive an award or a penalty to indicate the desirability of the event (Mitchell, 1997). The agent is not told directly what to do. The aim or goal of the agent is to maximize the total reward it will receive from the starting state. Some reinforcement learning algorithms assumes that the training data is available as real-valued reward signals given for each state-action transition. The training data is very seldom organized in such a way, so researchers have devised a set of algorithms that can handle having the reward and penalties given out at the end of the learning experience. For example an agent may play an entire round of a given board game and be given a reward for winning the game at the end. The agent then has the challenge of determining which of the actions in the sequence are to be credited with producing the reward .An algorithm designed to solve this problem is the Q-Learning algorithm, which is a popular algorithm for reinforcement learning, learns the agent an evaluation function Q(s, a). The evaluation function Q(s, a) is meant to determine the highest expected reward the agent can get when performing action a on a state s (Mitchell, 1997).

A problem often encountered in reinforcement learning is the problem of exploration. There is a tradeoff to be made between exploring new unexplored game states and exploiting state-action pairs already known to yield high rewards (Mitchell, 1997).

2.1.3.1.3 Artificial Neural Networks

Human beings use neurons to collect, process, and disseminate electrical signals in our brain. The field of artificial intelligence has taken inspiration from biologists and neuroscientists who have thought that the humans information processing capacity emerge from networks of these neurons (Russel & Norvig, 2003). The most common form of neural network is the feed-forward network. In a feed-forward network the information is fed forward through the layers as described below.

Neural networks consists of nodes connected by directed links (Russel & Norvig, 2003). Each link has a weight associated with it, the weight determines the strength and sign of the link (Russel & Norvig, 2003). Illustrated in figure 2.1 one can see a simple neural network. This network has an input layer with three neurons, a hidden layer with four neurons and an output layer with tow neurons. This network is called a Multilayer Feed-forward Neural network because the information is fed from the input layer on the left through the network and the output is given by the output neurons on the right.

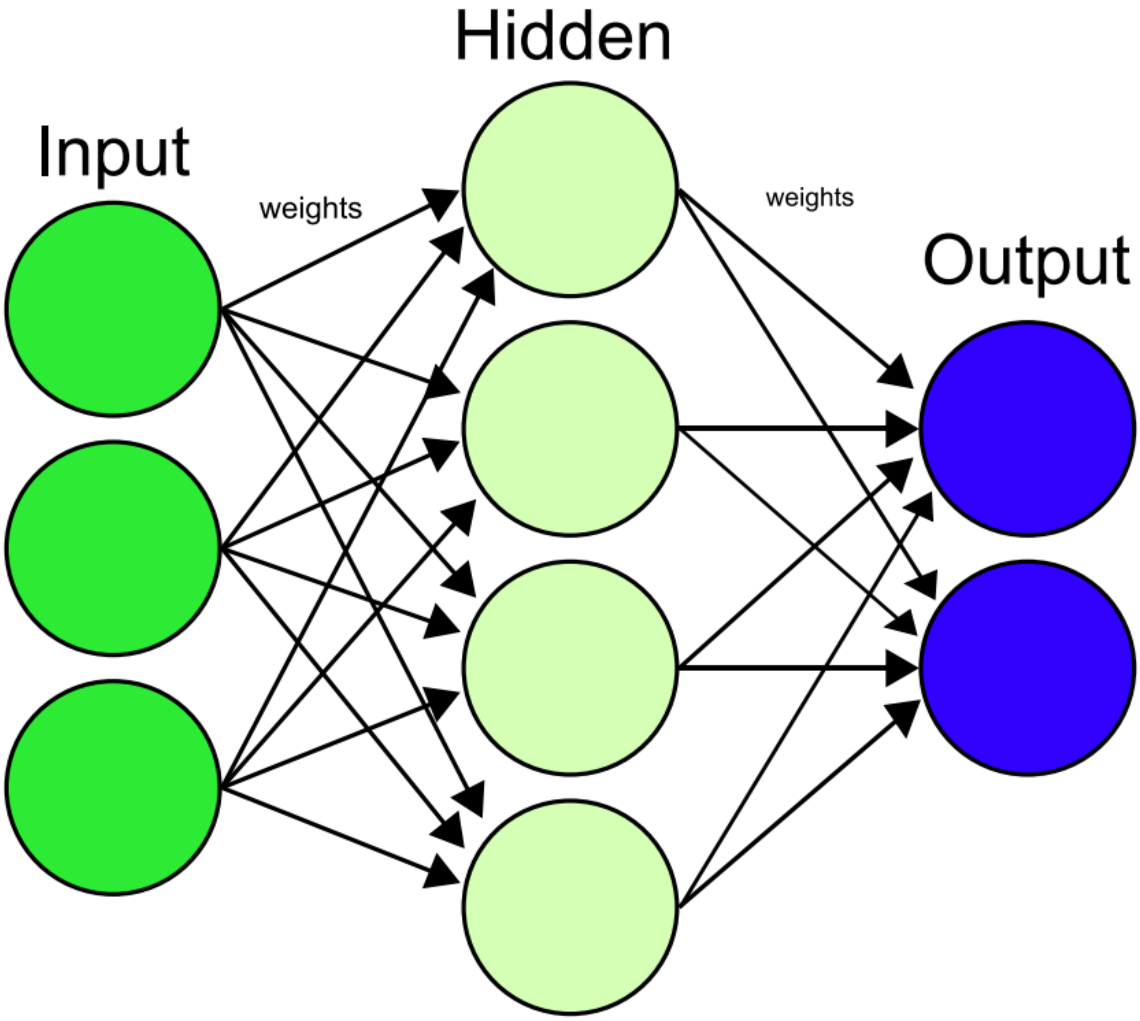


Figure 2.1: An Example feed-forward artificial neural network

A neuron first computes a weighted sum of its input. It then applies an activation function to this sum to derive the output (Russel & Norvig, 2003). If the function deems the input to be “right” it will output a number close to one, or zero otherwise. The activation function needs to be non-linear, in order to prevent the network from becoming one simple linear-function. An illustration of how a feed-forward neuron functions can be seen in Figure 2.2

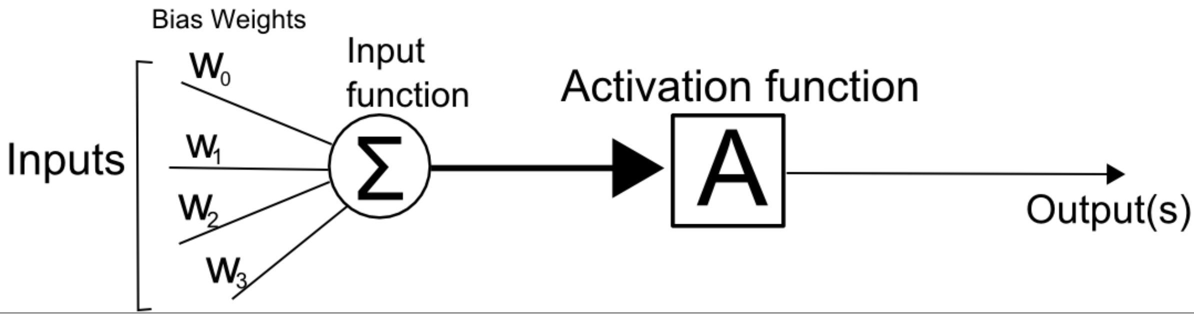


Figure 2.2: A feed-forward neuron

2.1.3.1 The Backpropagation Algorithm

The Backpropagation algorithm learns the weights for a multilayer neural network (Russel & Norvig, 2003). When we have training data that consists of input and output pairs, then we can calculate the error of an output neuron. The learning problem then faced by the Backpropagation algorithm is to backpropagate the error from the output layer to the hidden layers The algorithm uses the calculated back-propagated errors to adjust the weights. There are two approaches a backpropagation algorithm can take. The algorithm can adjust the weights for every input-output pair, or it can calculate an accumulated error for all the input-output pairs. Because the result of the backproagation algorithm will never be 100 percent perfect the algorithm needs to have one or more stopping criteria defined. The stopping criteria can be number of iterations (epochs), a satisfactory low error rate. Backpropagation is the most common algorithm for Artifical nerual networks, although many others have been proposed (Mitchell, 1997).

2.1.3.2 Appropriate problems for Artificial Neural Networks

Tom M. Mitchell (1997) lists some characteristics for problems that can be appropriately solved by artificial neural networks

- Instances are represented by many attribute-value pairs.
- The target function may be one or several discrete or real values.
- The training set may contain errors
- Fast evaluation of the trained target function required
- Acceptable with long training time

- The ability for humans to understand the target function not a requirement

2.1.4 Artificial Intelligence in games

The concept of artificial intelligence in games has been a concept since the term artificial intelligence was first coined by Alan Turing in the 1950s. With Turing encouragement Christopher Strachey wrote the first artificial intelligence program, and it was a player for checkers (Copeland, 2000). The first chess-playing program ran in November 1951, and was created by Dietrich Prinz (Copeland, 2000). Alan Turing was one of the first to mention that games could be used to benchmark an AI systems performance and intelligence (Copeland, 2000). The commercialization of the computer, and as a result the commercialization of video games, has led to an increased interest in game AI research. In 2001 the United States computer games industry business volume was higher than the one of the film industry. Digital environments are free of noise and are thus deterministic (Kleiner, 2005). Compared to the real world this makes making digital only artificial intelligence systems a lot easier. The gaming industry also allowed the hardware industry to grow at an exponential rate.
In 2000 Steven Woodcock completed a poll at the roundtable for game AI developers at the 2000 Game Developers Conference (GDC) (Woodcock, 2000). Comparing the results to the previous years he came up with the graph shown in figure 2.3.

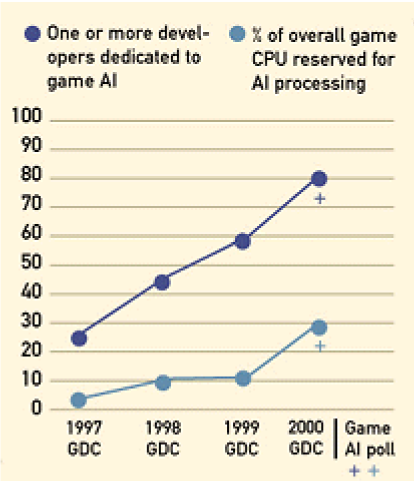


Figure 2.3: Poll from AI Round Table, GDC 2000

We can see that nearly every 80 percent of the developers reported that one or more developers worked dedicated to AI on either a current or previous project (Woodcock, 2000).

While one third reported to having two or more developers dedicated full time to AI (Woodcock, 2000). Woodcock looks positively to a future of game AI both in the industry and the academia considering the field is getting more developer time and cpu time dedicated to it in the industry (Woodcock, 2000).

2.2 Emotions

Emotions are one of the most important aspects of being human. Despite this there are very large discrepancies between definitions of how emotions work, and which emotions are important. Ortony, Clore and Collins (1988) define emotions as : "[...] valenced reactions to events, agents or objects, with their particular nature being determined by the way in which the eliciting situation is construed" (Ortony, Clore, & Collins, 1988).

2.2.1 Defining basic emotions

There are many emotion theorists who argue that some emotions have a different status than others, but few of these theorists can agree on which emotions are basic and which are not. Ortony et al (1988) claims that there are as many opinions about the number of basic emotions as there are opinions about their identity (Ortony, Clore, & Collins, 1988). There are some advantages to considering some emotions as basic emotions. Importantly for artificial intelligence is that the entire domain of emotions could then be described in terms of basic emotions. Picard argues that the obstacle created by "[…] the lack of agreement on whether there are basic emotions or continuous spaces of emotions" are not insurmountable (Picard, Affective Computing, 2000). Ortony and Turner (1990) conclude that we probably will never have an agreed upon criterion of the basicness of emotions. Despite this Ortony et al (1990) agree that it is viable as a research strategy to classify emotions in a certain way (Ortony & Turner, 1990). This view is supported by Picard who creates a good argument for "fuzzy" categories meaning that an emotion can belong in more than one category at once.

2.2.2 Affective computing

The project described in this paper falls under the field of affective computing. Affective computing is the study and development of systems and devices that can recognize, process, and simulate human emotions. The machine should interpret the emotional state of humans

and adapt its behavior to them, giving an appropriate response for those emotions. This is where tools like the Emotiv Epoc Headset can come into use, as discussed in the section 2.2.2.3. Modeling emotions is hard without real and reliable data on the humans' emotions; the Emotiv Epoc Headset helps combat this issue. There is a good amount of research and literature available in the field of affective computing, and there is a growing amount of work being done in the field. In 2000 Rosalind W. Picard wrote in her book that "The latest scientific findings indicate that emotions play an essential role in rational decision making, perception, learning and a variety of other cognitive functions" (Picard, 2000). R. W. Picard came to the conclusion that if we want computers to be genuinely intelligent, to adapt to us, and to interact naturally with us, then they will need the ability to recognize and express emotions, and to have what has come to be called "emotional intelligence" (Picard, 2000). Her book proposes just that, that we give computers the ability to recognize, express and in some cases "have" emotions (Picard, 2000). Picard pulls up an example from psychology where Damasio's patients have frontal-lobe disorders, affecting a key part of the cortex that communicates with the limbic system (Picard, 2000).This disorder results in the patients displaying a lack of emotions, and appearing unusually rational (Picard, 2000). As an example Picard (2000) mentions a patient, named Elliot, who seems unable to learn the links between dangerous choices and bad feelings, so he repeats bad decisions repeatedly instead of avoiding them (Picard, 2000). Picard argues that current artificial intelligence systems created so far display the same faults found in patients like Elliot (Picard, 2000). Artificial intelligence systems are coded with a large set of rules, which gives them good knowledge within an area, but AI systems are still not very good at making good decisions.
Picard suggests that computers having emotions is not the only part of making better artificial intelligence systems, humans interaction can greatly benefit if artificial intelligence systems can recognize the humans emotions (Picard, 2000). The example used in the book is from a learning situation. The learning situation, which is greatly improved when the subject is having fun and is engaged, can be compared to a gaming situation, where it's about having fun and being engaged.

2.2.2.1 The OCC-model

In order for an artificial intelligence agent to have emotions a model about how emotions are generated and how emotions affect decision making needs to be simulated. Ortony, Clore and Collins propose a model in which emotions are defined as valenced reactions to events, where

"[...] the winners and losers are reacting to the same objective event. It is their construal of the event that are different." (Ortony, Clore, & Collins, 1988). The OCC-model focuses on what conditions creates an emotion, and ignores things like facial expressions and or body language.

Although the OCC-model mentions emotion types Ortony et al writes that the particular words have been chosen as suggestive labels for a given category in the model only (Ortony, Clore, & Collins, 1988). There are 22 categories of emotions in the OCC model. The emotions are categorized as a reaction to events agents or objects. In the OCC-model there are three basic classes of emotions (Ortony, Clore, & Collins, 1988):

- Reaction to events; being pleased vs. displeased.
- Reaction to agents; approving vs. disproving.
- Reaction to objects; liking vs. disliking.

A computer agent will only be reacting to conditions in its environment, which makes the OCC-model a very popular model for creating emotional agents. Because the OCC-model specifies that emotions are valenced reactions to events the emotions need to get a negative or positive value. The conditions an agent react to can be events, objects and other agents. The agent's emotional reaction to an event depends on his goals or desires. As an example consider two people playing Battleship, where player A hits the player Bs battleship. Player A may feel joy, while player B may feel anger or distress. The event is the same, but their construal of the event is different (Ortony, Clore, & Collins, 1988). The OCC-model also allows for emotions to trigger other emotions. For example, being frustrated over a long period of time may make the agent angrier.

2.2.2.2 Synthesizing Emotions

One of the articles used in the previous thesis by Carlson and Hellevang (2010), "A categorized list of emotion definitions, with suggestions for a consensual definition" by Kleinginna and Kleinginna (1981), attempts to compile definitions and skeptical statements from a variety of sources in the literature of emotion (Kleinginna & Kleinginna, 1981).They classify the definitions and statements into an outlines of 11 categories (Kleinginna & Kleinginna, 1981). This article is supported by Ortony and Turners article "What's Basic About Basic Emotions?". In this article Ortony and Turner (1990) discuss the concept of basic, primary or fundamental emotions (Ortony & Turner, What's Basic About Basic Emotions?, 1990). Lerner and Keltner discuss in their article "Fear, Anger, and Risk" how fear and anger influence judgment and decision making in human beings (Lerner & Keltner, 2001).

The book "The Cognitive Structure of Emotions" by Ortony, Clore, and Collins has been mentioned previously, but is important enough for this project to warrant its own mention. In their book they are primarily interested in the contributions that cognition make to emotions (Ortony, Clore, & Collins, The Cognitive Structure of Emotions, 1988). They assume that emotions are a result of the way situations are viewed by the subject (Ortony, Clore, & Collins, 1988). Both winners and losers are experiencing the same event, but their view of it is different (Ortony, Clore, & Collins, The Cognitive Structure of Emotions, 1988). Ortony et al wrote that they believe it is important for machines to be able to reason about emotions, for cooperative problem solving, natural language processing and planning (Picard, Affective Computing, 2000). Many researchers have found that the OCC-model lends itself well for use in artificial intelligence, even if the authors of the model did not have this specific use in mind when they wrote their book The Cognitive Structure of Emotions in 1988.

There are some disagreement among theorists regarding the emotions and their attributes. Appraisal theorists argue that the target of the emotion anger is an important attribute of the emotion, while other theorists will say that the emotion is more basic and easier to measure. The OCC-model has anger as an emotion that is a result of both displeasure and disproving (Ortony, Clore, & Collins, 1988). This means that everything that an agent or person finds displeasing also makes him angrier. Furthermore the person or agent experiencing anger can target it towards events (also referred to as outcomes), persons or agents, and objects. Attributing anger to each of the different target types will elicit different negative outcomes. Attributing the emotion to a person or agent results in disapproval, being angry with yourself will result in feeling shame, being angry with another person will result in a feeling of reproach towards that person. Displeasure is experienced when the emotion is targeted towards an event or an outcome, the greater the disapproval or displeasure, the greater the anger (Ortony, Clore, & Collins, 1988). As a counterpoint to the appraisal theorists some theorists argue that the cognitive causes of anger may only intensify the existing sources of anger. They also argue that pain, displeasure and undesirable conditions do not need attribution to agency or interpretation.

There also exists some controversy on the separation of the outcome-focused emotion frustration from the anger emotion which focuses jointly on outcomes and agency (Ortony, Clore, & Collins, 1988). The OCC-model distinguishes emotional reaction to negative events directly caused by a different agent or person from the emotional reaction to negative events .

Theorists have argued that frustration can make a qualitative difference to anger, by transforming frustration into anger. Frustration can also make a quantitative difference to anger, increasing the intensity of anger (Clore & Centerbar, 2004).There is also research which includes a more fine-grained account of both anger and frustration. In the OCC-model for example anger includes feelings of reproach, shame, disgust and also frustration (Ortony, Clore, & Collins, 1988). There has also been made a case for having frustration as its own emotion. Clore et al (2004) writes in their article that one's choice on how to include frustration depends on how one chooses to view emotions (Clore & Centerbar, 2004). In their paper Clore et al (2004) lean towards concluding that frustration only becomes anger when it becomes agency focused (Clore & Centerbar, 2004).

2.2.2.3 Affective wearables

Picard (2003) mentions an experiment with a "wearable computer" where Picard and her students attempt to see if a wearable computer can detect a person's emotions over a period of time (Picard, 2003). Picard (2003) found that eight emotions could be distinguished at levels significantly higher than chance, they developed pattern recognition algorithms that attained 81% classification accuracy (Picard, 2003). Picard asks how we can enable computers to better serve people's needs, adapting to each human being, instead of treating one like a fictional idealized user (Picard, 2003). Picard (2003) also makes a key point that humans are affected by emotions, even if they are not showing that particular emotion at that very moment (Picard, 2003).

Picard presents a chapter on "Emotion Synthesis", in her book from 2000. "We can expect computer emotions to play a role in giving computers these more human-like abilities, together with improving their skills for interacting with people" (Picard, 2000). Picard (2003) argues that as we construct emotional systems we need to consider emotional intelligence, teaching computers how to control their emotions, when and how to express them, and how to correctly and wisely recognize and reason about emotion (Picard, 2000). Creating what Picard refers to as emotional intelligence in the StateCraft engine was started by Carlson and Hellevang (2010) by using theories and ideas from Rosalind Picard. The StateCraft emotional module is developed with influence from the OOC-model presented by Picard, which she again pulls from the book "The Cognitive Structure of Emotions" by Ortony, Clore and Collins (Carlson & Hellevang, 2010) (Picard, 2000). The OOC-model was not intended to be used for emotion synthesis, but is useful for synthesizing cognitive emotions (Picard, 2000) .

The OOC-model groups emotions according to cognitive eliciting conditions (Picard, 2000). The model assumes emotions arise from valenced reactions to situations (Picard, 2000).

2.2.2.4 Emotiv Epoc

The neuro headset planned to be used in this project is developed by the Australian company Emotiv Systems (Emotiv, 2011). The only current product of Emotiv Systems is the Emotiv Epoc neuro headset and its Software Development Kit. Emotiv Systems was founded by four scientists and executives; Professor Allan Snyder, chip-designer Neil Weste and technology entrepreneurs Tan Le and Nam Do (Emotiv, 2011). "The technology, which comprises a headset and a suite of applications, allows computers to differentiate between particular thoughts such as lifting an object or rotating it; detect and mimic a user's expressions, such as a smile or wink; or respond to emotions such as excitement or calmness" (Emotiv, 2011).
The Emotiv Epoc neuro headset consists of 14 saline electrode sensors for EEG (electroencephalography) readings. It also has a gyroscope which can measure movement along two axis. The device can also detect and categorize emotions into a variety of different categories of emotions through its affective suite (Emotiv, 2011).
The Emotiv Epic Software Development Kit comes with a three different modules. The Expressive Suite can detect facial expressions such as smile, wink, grin, laugh, and more. A suite called the Affective Suite comes with the Software Development Kit and allows one to get an image of the players' emotions, and this is the suite which will be most relevant for this research project. The Cognitiv Suite lets one train the Software Development Kit up so it can detect more detailed thoughts such as push, pull, rotate in different directions, and more. The Emotiv Control Panel which comes with the Software Development Kit and the Emotiv Epoc neuro headset lets one train up the SDK for different users very easily. It also gives a graph of players' emotions, which is similar to what I'm planning to achieve with a data analysis tool.
There has been some research done already that uses the Emotiv Epoc headset. One of them is a project by Azcarraga et al. titled "predicting academic emotion based on brainwave signals and mouse click behavior" (Azcarraga, et al., 2011). In this project Azcarraga et al asserts that academic emotions such as confidence, excitement frustration and interest may be predicted based on brainwave signals. Their paper looks at a case of Intelligent Tutoring Systems (ITS) that interact with the student through a computer that acts like a human teacher (Azcarraga, et al., 2011). By the help of various sensor signals from the mouse and an EEG headset Azcarraga et al wants to create an affective tutoring system, that can recognize and adapt to

the users affective state (Azcarraga, et al., 2011). Based on the data given to them from the brainwave signals and the mouse behavior data they try to predict and classify academic emotions. Azcarraga et al had twenty-five computer science undergraduate students use the intelligent tutoring system while wearing an EEG sensor. Data about the students’ mouse behavior, such as mouse clicks, click duration and movement, where captured and stored in log files (Azcarraga, et al., 2011). The students were also presented with a window for self-reporting their own emotions every 2 minutes. In this window the students reported intensity for the emotions confidence, excitement, frustration and interest with a value from 0 to 100 using a sliding bar . After creating six different datasets based on the percentage of feature, and balancing them by ensuring that there were the same number of instances for each emotion, Azgarraga et al classifies the modality of each emotion. In addition the classification included whether it was brainwaves or mouse, or a combination (Azcarraga, et al., 2011). In the end the authors conclude that the academic emotions (confidence, excitement, frustration and interest) may be predicted based on brainwave signals. Prediction rates based on brainwave signals only showed Azgarraga accuracy rates of 54% to 88% (Azcarraga, et al., 2011).

2.2.2.4.1 Affective Detection Details

The affective suite reports real time changes in the emotions experienced by the wearer (Emotiv, 2011). The detection values looked for are universal brainwave characteristics, but after extended use the detection will learn from individual users values and improve the accuracy for that user. The affective suite offers a number of emotions that can be observed in a universal way.

Excitement is reported in two forms; Instantaneous and Long-term excitement. Instantaneous excitement is a feeling of physiological arousal or awareness. A range of physiological responses are used to characterize excitement. These responses include pupil dilation, eye widening, sweat gland stimulation, heart rate and muscle tension increases, and digestive inhibition (Emotiv, 2011). The output is scored after how great the increase in physiological arousal is. The Instantaneous excitement is tuned to give a score on changes over time short time periods (seconds), while Long-term gives a score over a longer time period (minutes).

Engagement is described as alertness and attention towards task-relevant stimuli (Emotiv, 2011). Engagement is characterized by increased physiological arousal and beta waves along with attenuated alpha waves (Emotiv, 2011). **Boredom** is reported by the Emotiv headset as

the opposite of **Engagement**, but users have sometimes reported that this does not always correspond to the experience of boredom (Emotiv, 2011). **Engagement/boredom** is scored by how great the attention, focus and cognitive workload is (Emotiv, 2011).
Frustration is not described by the Emotiv Software Development Kit user manual but is still used in this project, and in other projects such as the project by Azcarraga et al (Azcarraga, et al., 2011).

2.2.3 Emotions and their role in human decision making

In order to model the players' emotion correctly one will also need literature from research on how emotions influence humans decision making and behavior. Because the game only has the behavior of the agent as an outward indicator of its decision making the behavior and decision making will be the focus of this chapter.
Loewenstein and Lerner assert that"[..] immediate emotions often drive behavior in directions that are different from those dictated by a consequentialist evaluation of future consequences". The immediate emotions can directly or indirectly impact the decision making, or alter the decision makers expectation of the probability or desirability of future events (Loewenstein & Lerner, 2003).
The findings of Loewenstein and Lerner (2003) are in line with what Roman V. Belavkin concludes in his paper "The Role of Emotion in Problem Solving". Belavkin investigated how the emotion controlled changes to the motivational states influence information processing. It is also shown that the dynamics corresponds to otimisation methods such as best-first search and simulated annealing (Belavkin, 2001). Belavkin concludes that emotions in general contributes to problem solving where positive emotions increase motivation and confidence, and negative emotions can help the decision maker overcome possible problems . It was found that arousal, motivation and confidence changed during the problem solving when emotions such as frustration and joy are experienced (Belavkin, 2001).

2.2.4 Emotional agents

There are two objectives to implementing Emotions into artificial intelligence agents; making the agents more believable, or improving or changing the agents' decision making.

2.2.4.1 Believable agents

The article "The Role of Emotion in Believable Agents" discusses how artificial intelligence researchers can learn from the work of artists who have explored the idea of believable characters (Bates, 1994). Bates lists three important points remembered by animators when creating believable agents:

1. A clearly defined emotional state at each moment. This makes the viewer able to see distinct emotions in a character.
2. The actions of the character reveal its emotions. The characters emotional state is clearly defined, so it's thinking and thus actions must also be clearly influenced by the characters emotional state.
3. Give the user time to grasp the emotional state. Use time to establish the emotion and present it to the users. Exaggeration and toning down of other things can get the user to notice the emotion faster or more strongly.

To get the first point covered Bates (1994) chose to use the OCC-model in order to make the agents experience valenced emotions based on events in their environment (Bates, 1994). For the second point Bates (1994) defined behaviors for each emotion included from the OCC-model (Bates, 1994).

In the paper "Emote to Win: Affective Interactions with a Computer Game Agent" Kim et al (2004) introduce a game interface that is based on affective interactions between a player and a computer pet. The basic idea of the game presented is to elicit certain reactions of the pet via appropriate emotive user behavior (Kim, Bee, Wagner, & André, 2004). Kim et al (2004) propose a system where a virtual pet is affected by the owners' emotional state (Kim, Bee, Wagner, & André, 2004). The emotion state of the user is read by a sensor that can measure skin conductivity, heart rate respiration and muscle activity, in addition to a speech input analysis (Kim, Bee, Wagner, & André, 2004). Kim et al (2004) divide their game environment into components: recognizing emotions from bio signals and speech, fusing the results from input, and determining and animating the behavior of the pet (Kim, Bee, Wagner, & André, 2004). The underlying emotion model Kim et al follows characterizes emotions in terms of arousal or valence (Kim, Bee, Wagner, & André, 2004). Kim et al (2004) attempt to recognize anger with negative valence and high arousal, calm with positive valence and arousal low, sad with negative valence and arousal low and happy with positive valence and arousal high (Kim, Bee, Wagner, & André, 2004).The four subjects were presented with videos in order to get data on the emotions Kim et al wanted to recognize. The virtual pet maps input about the users' emotional state onto facial and body behavior (Kim, Bee, Wagner, & André, 2004). Kim et al found that both affective speech and bio physiological feedback can be integrated into a computer game (Kim, Bee, Wagner, & André, 2004). "A Cognitive Psychological Approach to Gameplay Emotions" by Bernard Perron studies the appraisal and action dimensions of emotions arising from game play, from a cognitive psychological perspective (Perron, 2005). The emotion of "interest" is found to be important in film viewing, and thus also in story-driven games (Perron, 2005). Perron characterize some prototypical emotions seen in gameplay (Perron, 2005):

- Interest a tendency to pay attention, observe and understand a situation.
- Enjoyment is a mixed reaction which makes the person want to interact and prolong the game.
- Worry makes the person focus on an objective.
- Fear makes the person flee, run away or straight out avoid danger
- Anger is seen as an agnostic tendency by Perron (2005). It is used to regain control of a situation, with the help of aggression.
- Frustration shows some of the same agnostic tendencies seen in anger. Behaving short tempered.

2.2.4.2 Agents using emotions in decision making

Magy Seif El-Nasr and Majorie Skubic (1999) wrote an article titled "A fuzzy emotional agent for decision-making in a mobile robot" (El-Nasr & Skubic, 1999). In the article they explore how to use of emotional agents in the decision-making process of a mobile robot (El-Nasr & Skubic, 1999). El-Nasr et al chose to use a fuzzy model of the emotions in order to capture the inherent uncertainties. The agent makes decision based on environmental conditions and a set of emotional states; fear, pain and anger (El-Nasr & Skubic, 1999). To facilitate the decision-making process El-Nasr et al decides to use a framework based on the Intelligent Agent (IA) framework. In the model used the expectation levels of the agent determines the emotions and the emotion intensity. Emotions can both cause the agent to modify its goals and cause the agent to take actions that are based solely on the emotional state (no environmental inputs required) (El-Nasr & Skubic, 1999).

El-Nasr et al develop an algorithm for the agents' decision making. The algorithm normalizes three sources of input; brightness level, sound level and if the agent is alone, physically damaged or blocked (referred to as the agents state) . The algorithm then evaluates the expectations according to the inputs given. The environment also supplies the input El-Nasr calls 'stimulus'. Stimulus is an event or object that is used to calculate the expectation and desirability of a stimulus. Stimulus is more desirable if they can fulfill a goal. Based on both the expectation and desirability values the algorithm infers the emotional state. Once the emotional state is calculated the algorithm chooses an emotion based on a priority system and the emotions intensity (El-Nasr & Skubic, 1999). The chosen emotion goes into a behavioral system and according to the emotion's intensity and the agents' state an action will be recommended. The chosen emotion will have its intensity decreased, while the emotions not chosen will be sent back into the system.

Velásquez presents a neuropsychology inspired approach to the study of emotions and decision-making. In his paper "Modeling Emotion-Based Decision-Making" Velásquez proposes a framework for Emotion-Based Control (Velásquez, 1998). The model proposed consists of five different modules:

- Perceptual systems get information from the world and provide the emotional and behavior systems with stimuli and objects
- Drive systems are motivational systems that 'drive' an agent into actions, for example the agent can have a Hunger drive and the agent will be more inclined to obtain food (Velásquez, 1998).
- Emotional Systems represent various emotional responses, such as Fright, Fear, Terror and Panic (Velásquez, 1998). The cognitive emotion releases are learned by the agent through its lifecycle in the world (Velásquez, 1998). In addition fast primary emotions, emotion blends, and emergent emotions are modeled (Velásquez, 1998). The emotional systems also contain modules for mood and temperament, which allows Velásquez to create grumpy or joyful agents (Velásquez, 1998).
- Behavior Systems are responsible for choosing how to respond to an event. Behavior systems may inhibit or excite each other (Velásquez, 1998).

The emotions act as the main influence on how behaviors are selected (Velásquez, 1998).

Chapter 3

3 StateCraft and the Emotion Synthesizer

In 2006 Helgesen and Krzywinski implemented a computer version of the board game Diplomacy. They decided to name the game StateCraft. In the StateCraft game autonomous intelligent agents can play against other agents or human players. The StateCraft game has been worked on in iterations. In 2008 a Personality module for the autonomous agents in StateCraft was created and evaluated by Jensen and Nes. Carlson and Hellevang (2010) expanded the autonomous agents further with an Emotion module and a Prisoner’s Dilemma module. Carlson and Hellevang designed the Emotion module based on data gathered from four interviews from one game of the board game version of Diplomacy.

3.1 Diplomacy

Diplomacy is a strategic social multiplayer board game developed by Allan Calhamer after the Second World War (Calhamer, 1974). Diplomacy is set in Europe just before the First World War, there are seven great powers in the game. The seven powers (Russia, The United Kingdom, France, Germany, Italy, The Ottoman Empire and Austria-Hungary) seek to control Europe.

The game board is a map of Europe (plus some parts of the Middle-East and Asia and some parts of North Africa) divided into 75 land, water, or coastal provinces. Each power can control build and command armies and fleets. Army units are used on land, or coastal areas. Fleet units can occupy water or coastal areas. Armies and fleets can be ordered to move, hold position, or assist both friendly and opposing units. In addition fleets are able to convoy armies from one coastal province to another over a sea area.

At the start of the game every nation starts with a set of provinces (considered the nation's home provinces), armies, and fleets. A set of the provinces also contain a supply center. The number of supply centers dictates how many units a player can have on the map. For each supply center controlled a player can control one unit (fleet or army). If a player wants to build additional units he must first seek to control additional supply centers. If a player has less supply centers than he has units he has to destroy one unit. Only a nation's home provinces can build units, occupied provinces cannot be used to build units. Only one unit can be in a province at a time. In order to win the game a player needs to be controlling 18 supply centers.

When a player invades a province there are 5 different scenarios that can take place (Carlson & Hellevang, 2010):

Scenario 1: A unit moves into an unoccupied province, with no other attackers trying to occupy it. The unit then occupies that province.

Scenario 2: A unit moves into a province which is occupied by an enemy unit. This leads to a standoff and the unit has to retreat. The enemy unit keeps control of the province.

Scenario 3: A unit moves into a province, with support from a friendly unit. The province is occupied by an enemy unit. The enemy unit then has to retreat to a friendly province, or is disbanded. The attacking unit gains control of the province.

Scenario 4: A unit moves into a province, with support from a friendly unit. The province is occupied by an enemy unit, the enemy unit is supported by another enemy unit. This leads to a standoff like in scenario 1, and the attacking unit has to retreat.

Scenario 5: A unit moves into an unoccupied province. An enemy unit also moves into the unoccupied province from a different province. This leads to a standoff, and both units have to retreat.

Here one can note that there is no element of randomness in the game, which makes the combat system very interesting for AI purposes.

Diplomacy is divided into four seasons (spring, summer, autumn, and end-of-year winter), two seasons for action and two seasons for negotiation. In the action rounds each player is able to give orders to his units. The orders can be moving, supporting or convoying units. Each of the action rounds are preceded by a negotiation phase. In the negotiation phases of the game the players negotiate amongst themselves and form alliances. In the winter and summer rounds the orders are made public to all and set into action. This means that the promises made are kept or broken in the winter or summer.

3.2 StateCraft

Carlson and Hellevang (2010) aimed to improve the user experience in the existing StateCraft game made by Helgesen and Krzywinski. Carlson and Hellevang (2010) add an Emotion module and a Prisoner's Dilemma module to the StateCraft game engine. They aim to study whether an agent equipped with emotions will enhance the user experience (Carlson & Hellevang, 2010). The thesis focuses on simulating emotions in an agent so that it appears more human-like, with the goal of increasing the player's game experience (Carlson & Hellevang, 2010). Their design and implementation of the Emotion module is derived from a player study they conducted. Seven players were gathered to play the board game, and four of them were interviewed about their emotions afterwards. They let the Ortony Clore Collins-model (OOC) developed by Ortony et al in 1988 combined with the information collected in the player study form the foundation for the Emotion module (Carlson & Hellevang, 2010).Carlson and Hellevang present the OOC model as one of the most popular for synthesizing emotions (Carlson & Hellevang, 2010). The way they use the OOC-model to implement their own emotional model, specific to the StateCraft and Diplomacy game play is described in full in their thesis paper. Events are considered things that can happen, and the agent's reaction depends on his goals (Carlson & Hellevang, 2010). As an example Carlson and Hellevang (2010) argues that since the main goal of the StateCraft agent is to gain 18 provinces, losing a province would cause him distress and displease him (Carlson & Hellevang, 2010).They take inspiration the Three-Layer Architecture proposed by Aaron Sloman book in their project (Carlson & Hellevang, 2010).They conclude that a Three-layered approach is needed to successfully implement an agent with emotions (Carlson & Hellevang, 2010).

3.2.1 The Three Layered Caenus Architecture

Diplomacy is a game with two important aspects that the artificial intelligence agent has to handle. One aspect is the social aspect which involves a non-deterministic, dynamic and continuous social environment. In addition the agent has to handle the game itself. Helgesen and Krzywinski observed that the players in Diplomacy engaged in three different activities; observing the game state, considering the next move and negotiation with other players (Helgesen & Krzywinski, 2006). Helgesen and Krzywinski used these observations to argue for a three layered architecture for their implementation of a Diplomacy AI agent, and they tie each of the activities up to a layer in the architecture.

The three layers described by Helgesen and Krzywinski (2006) are:

The **Operational layer** focuses on single pieces and their opportunities

The **Tactical layer** focuses on how all the pieces can combine their efforts

The **Strategic layer** focuses on diplomatic negotiation and long-term planning

The layers operate concurrently, and have their own internal computation mechanism for processing received input (Helgesen & Krzywinski, 2006).

3.2.1.1 Operational layer

The operational layer is triggered by a new game state and starts the module that discovers all the possible moves a player can perform on the current game state (Helgesen & Krzywinski, 2006). The operational layer does not try to rank or order the valid moves it discovers in any way. The operational layer reacts purely to the new game state (Helgesen & Krzywinski, 2006).

3.2.1.2 Tactical layer

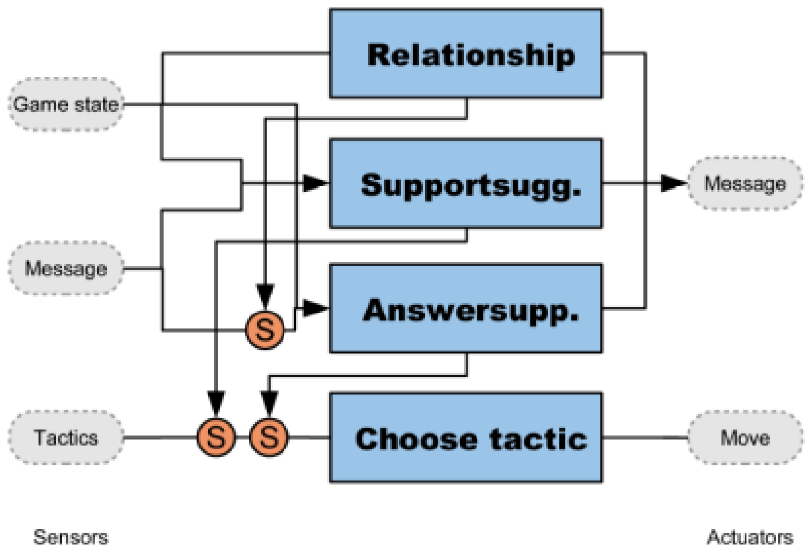
The tactical layer receives the game state and a list of valid moves from the operational layer and uses this information to generate tactics. A tactic is a decision for every unit a player controls (Helgesen & Krzywinski, 2006). The tactical layer is also responsible for ranking the tactics by value; the tactics are given a high value if considered good, while poor tactics are given a lower value. These values are based upon several heuristics. For example a tactic which involves controlling a supply center would be valued higher than one that did not.
The tactics are given two different values, calculated in the Valuator class. The two values are a tactic's potential value and a tactic's factual value. The potential value is the value of a tactic without considering the pieces of the competitors. Potentially good moves can be conquering a supply center or moving a piece to a better position. The factual value of a tactic represents the probability of whether or not a move is a success. If a tactic is 100% likely to succeed its potential and factual values would be equal. Helgesen and Krzywinski (2006) designed several heuristics to calculate these values.

3.2.1.3 Strategic layer

The strategic layer is tasked to communicate with other players, make plans and decide what action to perform. This makes it the most complex of the three layers. The implementation originally consisted of 4 modules from Helgesen and Krzywinski (2006). The strategic layer implements the Subsumption architecture, which was originally designed to control robots in a real-world environment. The implementation originally consisted of four modules (see figure 3.1)

- The **ChooseTactic** module, chooses the tactic the agent performs, based on its factual value and a little randomness
- The **SupportSuggestor** module looks for game states where an opponent can contribute with support. It is also the module that sends the support request messages.
- The **AnswerSupportRequest** is the module that receives support requests from the other players and decides if the agent should answer yes or no, based on criteria such as relationship to the other player and randomness.
- The **Relationship** module keeps track of the relationship to other countries and adjusts this based on opponents' actions. Relations can be Friend, Neutral or War.
- The **Planner** evaluates the agent's position in the game and selects long and short-term goals.

The Subsumption architecture allows layers to suppress or inhibit the input and/or output of lower layers. The behavior of the agent changes with each layer that gets added to the model. The modules are able to override input to other modules by acting as suppressors (the orange S-symbols in figure 3.1), or override output from modules by acting as inhibitors (Helgesen & Krzywinski, 2006).



Inhibitors are not depicted in this picture, as they were not used for any of the modules in the original version of StateCraft

Figure 3.1: Architecture of the Strategic Layer

(Helgesen & Krzywinski, 2006)

3.2.1.4 TacticTree

The TacticTree data structure, invented and implemented by Carlson and Hellevang (2010), is used to generate and represent all the tactics for a given agent. The number of tactics an agent can have is determined by the number of legal permutations of operations. This can mean that an agent can have thousands, hundreds of thousands or even millions of tactics. In Helgesen and Krzywinski’s StateCraft each tactic was represented as a list of operations, making generation of tactics a repetitive task where each tactic was cloned and changed marginally to accommodate the small change that separates one tactic from another . Carlson and Hellevang found that this had a huge impact on the time and memory consumption of the agent. In order to free up some time and memory used by the agent they introduced the TacticTree data structure. The TacticTree represents the tactics as a tree structure rather than as a flat list-based structure (Carlson & Hellevang, 2010).
The TacticTree constists of OperationNodes and each OperationNode contains a reference to an Operation-object. Each OperationNode has a reference to a parent node, another OperationNode or the root (Carlson & Hellevang, 2010). A path in the tree is semantically equivalent to a list-based tactic, where each element in the list is the same as an OperationNode in the TacticTree . This means that generating a Tactic no longer involves going through a TacticList that could be very big.

3.3 Emotion module

The Emotion module is a module developed for the StateCraft game by Carlson and Hellevang in 2010. The module has the purpose of researching whether emotions will affect the agents’ performance or the user's game experience (Carlson & Hellevang, 2010).

Carlson & Hellevang (2010) decided on the OCC-model in order to model and synthesize emotions in their module. A simplified version of the OCC-model has been implemented in related projects, and those projects showed that the OCC-model was fitting to synthesize emotions in agents (Carlson & Hellevang, 2010).

Carlson and Hellevang conducted a player study where 7 players were invited to play the board game. During the course of the game they were asked to describe their emotions, and how their emotions affected the game play. Based on the answers they created five emotions for the agents (Carlson & Hellevang, 2010):

- **Joy** is the reaction to an undesirable event.

- **Fear** is the prospect of undesirable events happening in the future.
- **Anger** is the distress one feels because of undesirable events combined with the reproach felt towards a person who were responsible for an undesirable event
- **Admiration** is the feeling towards a person who were responsible for a desirable event
- **Guilt** means feeling reproach towards yourself as a result of undesirable actions taken against an agent

Admiration replaces the emotion described as loyalty by the interviewees. Carlson and Hellevang (2010) argue that the admiration emotion in the OCC-model fulfills the same role that the interviewees describe as loyalty. The OCC-model does not include the emotion guilt either, but Carlson and Hellevang argue for including it on the grounds that it plays an important part in the domain of social board games (Carlson & Hellevang, 2010).

3.3.1 Emotion intensity

Emotions differ in intensity, represented by a value between 0 and 100. The emotions start out at a default of 0. For an emotion to affect the agent's decision-making it needs to exceed the threshold set for the particular emotion (Carlson & Hellevang, 2010). The emotions in Carlson and Hellevang's emotion structure have a different intensity value towards each player, except Joy which has a general intensity value. Joy and admiration can also have values down to -100. Negative joy represents the OCC-model distress, while negative admiration represents reproach (Carlson & Hellevang, 2010). The strongest emotion will suppress the other emotions.

To decide the values for each emotion Carlson and Hellevang defined game events that would change emotions intensity. It should also be noted that the anger emotion is a combination of distress and reproach, the negative sides of joy and admiration.

3.3.2 Affecting the agent's decision making

Based on the interviews Carlson and Hellevang conducted they defined how agent emotions affect its decisions:

- **Joy** will make the agent perform more risky moves, since it will give it the feeling of "being on a roll" (Carlson & Hellevang, 2010).

- If an agent has great **admiration** towards an opponent the agent is more likely to perform support orders as promised, and less likely to attack the opponent.
- **Anger** will decrease the chance of performing support orders for the opponent, and increase the chance of attacking the opponent.
- **Fear** will decrease the chance an agent has of attacking that opponent.
- **Guilt** will increase the chance of giving support to an opponent, and decrease the chance of attacking them.

3.3.3 Emotion module implementation details

The Strategic layer uses an implementation of the Subsumption system, with sensors to look for changes in the environment and actuators to act on the environment. The Emotion module receives game states from the server through the GameStateSensor, and SupportRequestMessages and AnswerSupportMessages through MessageSensor. The Emotion module performs actions by suppressing the input to ChooseTactic, in addition to inhibiting the output from the AnswerSupport module (Carlson & Hellevang, 2010).

3.3.3.1 EmotionSynthesizer

The main class in the Emotion module, EmotionSynthesizer, is implemented as a StateCraft module. StateCraft modules inherits the receive()-method, making it able to receive game states and messages from GameStateSensor and MessageSensor.

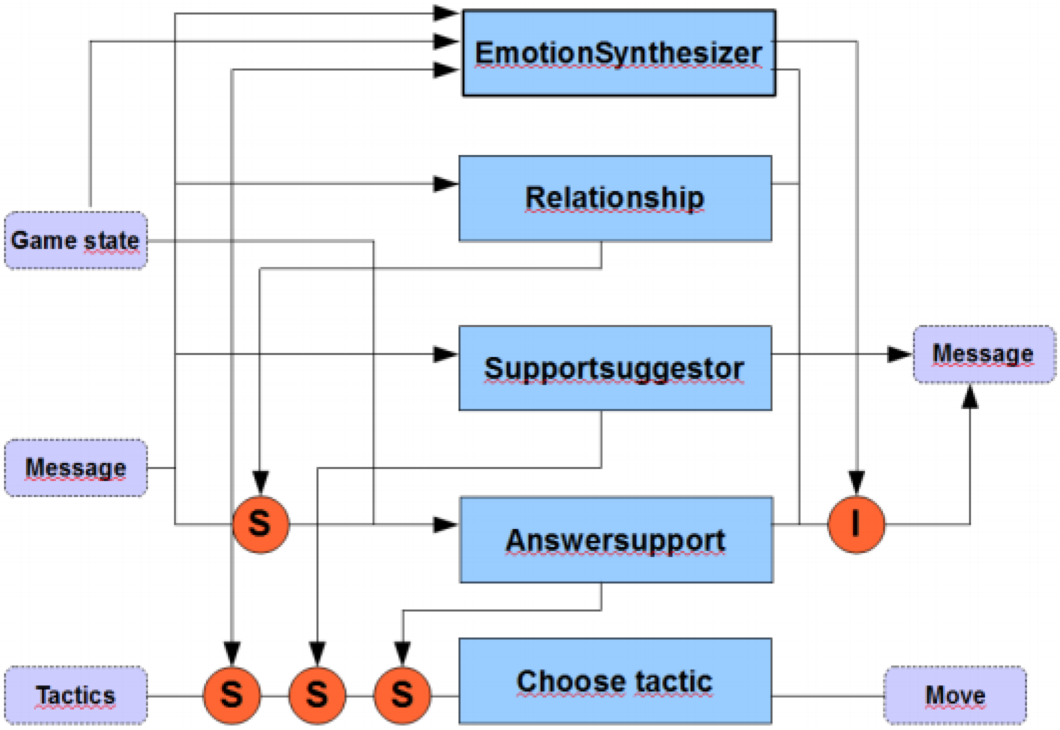


Figure 3.2: Emotion module in the strategic layer

(Carlson & Hellevang, 2010)

- **receive()** receives the GameState and the diplimatic messages. Each emotion changes its intensity based on the last round. The EmotionSynthesizer keeps track of the deals made last round when the SupportRequestMessages and AnswerSupportRequestMessages are passed through the receive() method (Carlson & Hellevang, 2010).
- **suppress()** is the method that calls the affectChoices()-method for each emotion in order to suppress the TacticList from the ChooseTactic module.
- **inhibit()** inhibits the outgoing AnswerSupportMessages. The messages are stored until next round so the agent can check if opponents kept their promises for support.

3.3.3.2 The Emotion interface

All implementations of the Emotion interface are required to inherit the following methods:

- **affectEmotion()** implements the rules defined for how each emotions intensity changes (Joy, Admiration, Anger, Fear, Guilt).

- **afffectChoices(TacticList)** is where the emotions influence the agent's decisions based on the emotions' intensities.
- **getValueFor(Country)** returns the emotion's intensity towards the specified country.

Chapter 4

4 Design and Development

This chapter contains detailed description of the systems implemented in order to use data from the Emotiv Epoc headset to improve the Emotion Module originally designed by Carlsen and Hellevang (2010). This chapter is divided into three sub-sections, one for each system developed for the thesis:

1. The Emotion Logger
2. The Emotion Learner
3. The StateCraft Emotion Module

Before the project started the idea of using a machine learning algorithm had already been discussed. When implementing the Emotion Logger it was clear that a machine learning algorithm would be the most practical and best solution. This realization made for the following figure (Figure 4.1) of the overall project:

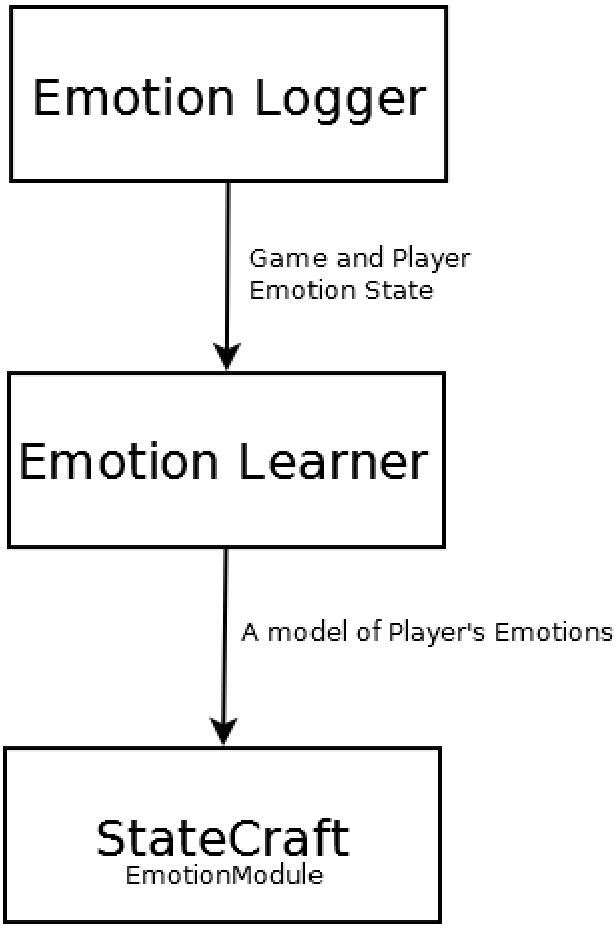


Figure 4.1: The three parts of the projects

4.1 The Emotiv StateCraft Emotion Logger

Generating the data for the AI lays down the basis for the following phases of the project. In this chapter there is an explanation of what data the Emotion Logger generates and how it gets access to the various data points.

This chapter will go through what data the CSV contains, and how it is obtained. The following chapters will go through how they are used.

4.1.1 The gathered data

The data generated by the Emotion Logger should be fitting to be used in a machine learning algorithm, meaning that it has to be structured in a way that is readable by a computer program. The data was saved to a file separated by commas, making it a comma separated values (CSV) file. A CSV-file stores tabular data in plain text. Each record is separated by line breaks, while each field is separated by a semi-colon. This means that the file is easily read by a computer program since it can be parsed by commas, and read by a human since it can be imported into a spreadsheet program.

4.1.1.1 The State of the Game

To be able to analyze what events triggers which emotions the Emotion Logger needs to log both the emotions from the headset and the game state which triggered that set of emotions. A game state consists of a number of relevant values saved into the fields:

- GameState - the season and year of the game state, e.g. Summer 1901
- Number of Provinces - an integer with how many provinces the player has
- Number of Supply Centers - an integer with how many provinces the player has
- Center Surplus - how many supply centers the player has versus how many armies and fleets he has (how much he can potentially build next build season)
- Number of Fleets - how many fleets the player has
- Number of Armies - how many armies the player has
- Occupied Neighbor - how many of the players provinces has an enemy unit in a neighboring province

- Supply Center Differences - a field for each country with the difference between the players number of supply centers and the enemies number of supply centers
- Supply Center Thieves - a list of countries who stole a supply center from the player this round
- Province Thieves - a list of countries who stole a province (without a supply center) this round
- Accepted received support requests - a list of support requests received this round that the player accepted
- Declined received support requests - a list of support requests received this round that the player declined
- Accepted sent support requests - a list of support requests sent this round that the receiver accepted
- Declined sent support requests - a list of support requests sent this round that the receiver declined
- Orders made - a field for each country containing what orders they made the last round

4.1.1.2 The State of the Player

In order to log the players' emotions, as mentioned previously, the Emotiv Epoc headset is used. The software development kit (SDK) provided is called Emotiv Education Edition SDK, and the version is version 1.0.0.4. The SDK is originally in C++, but because there are Java-bindings supplied with the SDK it is possible to use the SDK in the StateCraft java game.

The SDK supplies Java sample classes that show how to connect to and read information from the Emotiv Epoc headset. A Java package that allows the StateCraft game to read emotion data from the Emotiv Epoc headset was created. This package provides a class, named EmoStateLog.java, which in turn lets a thread be ran alongside the rest of the StateCraft game. The thread provides the following values on the players' emotional state:

- Excitement Short Term - a value between 0 and 1 describing the short term excitement of the player

- Excitement Long Term - a value between 0 and 1 describing the long term excitement of the player
- Engagement Boredom - a value between 0 and 1 describing how bored the player is
- Frustration Score - a value between 0 and 1 describing the frustration of the player
- Eye Brow Extent - a value between 0 and 1 describing the short term excitement of the player
- Smile Extent - a value between 0 and 1 describing how much he smiled this round

The data is always available at the end of a players turn to be written to a log file.

4.1.3 Emotion Logger Implementation details

4.1.3.1 Game State logging

The logger class called PlayerLogger.java is initiated with a reference to which country the player is playing currently. The country variable is used to dissect the GameState object which is received through the newGameStateReceived(GameState) method. From the GameState object one can extract a CountryState object for each country. The CountryState object for the players country gives us Number of Provinces, Number of Supply Centers, Center Surplus, Number of Fleets, Number of Armies and Number of Occupied Neighbor Provinces. From the received GameState the Supply Center Difference, Supply Center Thieves and Province Thieves are calculated.
In order to log the messages the player receives from the other players the messages are passed to the logger when they are received. The other players messages are received in the form of a SupportRequestMessage which contains information on who the sender is and what province the sender wants support to and which province the sender wants support from. The method also receives a RequestAcceptanceMessage which is the answer the player sends back to the opponent with an answer. The value of the RequestAcceptanceMessage will decide if the SupportRequestMessage is logged in the "Accepted received support requests" field or the "Declined received support requests" field. When the opponent answers one of the players requests for support the game receives a RequestAcceptanceMessage which is passed to the Emotion Logger. The data here is logged to either the "Accepted sent support requests" or the " Decline sent support requests" field.

```
		int state = 0;

		while (true) {
			// The current state of the EmoEngine
			state = Edk.INSTANCE.EE_EngineGetNextEvent(eEvent);

			// New event needs to be handled
			if (state == EdkErrorCode.EDK_OK.ToInt()) {

				int eventType = Edk.INSTANCE.EE_EmoEngineEventGetType(eEvent);
				Edk.INSTANCE.EE_EmoEngineEventGetUserId(eEvent, userID);

				// Log the EmoState if it has been updated
				if (eventType == Edk.EE_Event_t.EE_EmoStateUpdated.ToInt()) {
					Edk.INSTANCE.EE_EmoEngineEventGetEmoState(eEvent, eState);

				// Keep the smile extent if it is bigger than our last seen smile
					if(EmoState.INSTANCE.ES_ExpressivGetSmileExtent(eState) >
						smile) {
						smile =
	EmoState.INSTANCE.ES_ExpressivGetSmileExtent(eState);
					}

				}
			} else if (state != EdkErrorCode.EDK_NO_EVENT.ToInt()) {
				break;
			}
		}

		Edk.INSTANCE.EE_EngineDisconnect();
		System.out.println("Disconnected!");
```

4.1.3.2 Emotiv logging

The logger for the players' emotions is ran in its own thread alongside the rest of the program, called EmoLog. The class uses the interfaces and Dynamic Link Libraries (DLL) supplied by the SDK in order to connect to the Emotiv Headset through the Java Native Access (JNA) library. The Edk interface gives access to the Emotiv Epoc Headset by loading the DLL files and supplying methods to access the various data the Emotiv Epoc Headset can supply. In the EmoLog class the Edk supplies pointers to the EmoState and the EmoEngines locations in memory. Shown in the following code is the continually running code in the EmoLog class which handles event and error codes given by the Emotiv Epoc SDK.

This method does two things; keep track of the biggest smile and making sure there is a connection to the Emotiv headset.

Every time the Emotion Logger needs the emotion data to write to a new line of the log it will query the EmoLog thread for it. The method getEmotivState (see following code snippet) will return a string ready to be inserted into the CSV-file. The string returns contains Excitement Short Term, Excitement Long Term, Engagement Boredom, Frustration Score, Eye Brow

Extent and Smile Extent. The scores for the emotions are taken from the most recent data the EmoLog has available. Through the development of the logging module it was seen that the smile extent values were too sporadic if one used the latest available value. To solve this the biggest smile extent value seen since the last query is used.

```
/**
 * ExcitementShortTerm, ExcitementLongTerm, EngagementBoredom, Frustration, Eyebrow Extent,
* SmileExtent
 * @return csv string with emotiv player data
 */
public String getEmotivState() {
        String text = "";
        // ExcitementShortTerm
        text += EmoState.INSTANCE.ES_AffectivGetExcitementShortTermScore(eState)+";";
        // ExcitemntLongTerm
        text+= EmoState.INSTANCE.ES_AffectivGetExcitementLongTermScore(eState)+";";
        //EgagementBoredom
        text += EmoState.INSTANCE.ES_AffectivGetEngagementBoredomScore(eState)+";";
        //Frustration score
        text += EmoState.INSTANCE.ES_AffectivGetFrustrationScore(eState)+";";
        // EyeBrowExtent
        text += EmoState.INSTANCE.ES_ExpressivGetEyebrowExtent(eState)+";";
        //SmileExtent
        text +=smile + ";";
        smile = 0;

        return text;
}
```

When the player hits the next round button in the game the EmotionLogger will save all the gathered data from the game state to the CSV-file. At this point all the support requests are known, and it is the time the player reviews all his orders for that turn. This gives the best, most practical, fit for mapping game state to emotions. Once a game has been played the CSV-file is approved manually and renamed to the appropriate country. E.g. if a round as england was played for the first time the log file would be named england_1.csv.

4.2 The Emotion Learner

The next step of the project was to create a function that maps from game state to emotion state. To do this it was a natural choice to choose a machine learning algorithm.

4.2.1 Importing the Emotion Logger data

Because the training can take place outside of the StateCraft game a new project was created for the Emotion Learner. The first thing the Emotion Learner needs to do is read the CSV-files and put them into a machine readable format. Splitting the data in the CSV-file into the data structure java classes named EmotivState and GameState allows handling of the two states separately. The GameState class holds all the data relevant to the state of the game, which would be the input to the machine learning algorithm. Additionally an object of the GameState class holds a reference to the corresponding EmotivState object. While the EmotivState class holds all information relevant to the emotional state of the player, which would be the desired learned output of the machine learning algorithm.
Because the emotions found in EmotivState are undirected emotions there is little need to keep track of who does what to the player. Since all actions done to the player will increase or decrease the Emotiv emotions the GameState will simply quantify all the data if the data is not already in number format.

4.2.2 Training a learned function

As described in the chapter 2.1.3.1 there are some criteria to consider when choosing a machine learning algorithm. From reviewing literature, and from own experience, it was decided that artificial neural networks would be the most obvious choice. The learner has to learn a function which has to output several values, evaluate input quickly once trained, and be able to handle partially inaccurate training data. Long training times are also acceptable as long as it is able to act fast once implemented into the StateCraft game. Artificial neural networks are flexible, easy to experiment with and offer a wide variety of learning algorithms.

4.2.2.1 Artificial Neural Network Framework

Because this project explores a relatively new field there is a benefit in being able to try out different approaches to learning. There is not a known best practice for using machine learning with emotions in games. With this in mind it was decided that a framework would be largely beneficial. Using a framework would allow fast experimentation by using different combinations of algorithms and methods without having to implement these from scratch. If the framework is open source then it has the security of being reviewed and accepted implementations of the most popular and relevant algorithms.

The three major open source frameworks for artificial neural networks in java are:

- Encog (Heaton Research, 2012)
- JOONE (Joone Project on SourceForge, 2012)
- Neuroph (Neuroph on SourceForge, 2012)

In order to decide upon a framework for the project, a simple ANN was implemented using each of the frameworks. The Encog framework stood out as the easiest to use and is well documented. This is supported by a benchmark of Encog, Joone and Neuroph done on behalf of The Code Project (The Code Project, 2010). The conclusion from this benchmark is that Encog provides an easy to use API and superior performance (The Code Project, 2010). It is also worth noting that the Neuroph project and the Encog project collaborate, which speaks to the size of how many developers have a stake in the Encog project. The Encog project offers a wide variety of well documented features for creating and trianing artificial networks.

4.2.2.1.1 Training an Encog Neural Network

The Encog framework offers six different forms of propagation learning. Propagation training is, as described in chapter 2.1.3.1, a form of supervised training. In the case of the Emotion Learner the propagation learner will be given a GameState as input and an EmotivState as desirable output. The training algorithm iterates through until the desired error rate is hit, or until the maximum number of iterations have been done. The error rate is the percent difference between the desired output and the actual output (Heaton, 2011). Each iteration will go through the entire training set, and for each training element a forward and backwards pass is made (Heaton Research, 2012). The forward pass is used to calculate the output (EmotivState) and error for each training element. The backward pass applies the neural networks actual error to the derivative of the activation function to calculate the error gradient. The error gradient is then multiplied by the error. Which training algorithm one chooses defines how this value is used. The six training algorithms in the Encog framework are described and discussed briefly below:

- **Backpropagation** uses two parameters in addition to the gradient descent calculated above; learning rate and momentum. The gradient is multiplied by the learning rate, which slowly optimizes the weights to values that produces lower errors (Heaton, 2011). The momentum parameter is there to help the backpropagation algorithm get out of local minima. The backpropagation algorithm keeps track of the changes made to the weights last iteration. The changes from the previous iteration, scaled by the momentum parameter, are reapplied to the each new iteration (Heaton, 2011).

- **The Manhattan Update Rule** attempts to remedy the problem of the gradient descent used in Back propagation training often results in changes that are too big. The Manhattan Update Rule discards the magnitude but keeps whether the gradient is positive, negative or near zero. This magnitude is then used to decide how to update the weights. If the magnitude is near zero then the weight remains unchanged, if the magnitude is negative then the weights value is decreased by a specific amount and vice versa for positive magnitude. The specific amount is defined by a constant one provides before the algorithms starts.
- **Quick Propagation** was devised by Scott E. Fahlman and published as a paper in 1988 (Fahlman, 1988). The algorithm considers the slope of the error curve. If the slope of the error curve is less than that of the previous one, then the weight will change in the same direction. The QPROP algorithm takes a learning rate parameter to prevent the change from becoming too big.
- **Resilient Propagation Training** The resilient propagation training algorithm (RPROP) requires no parameters, and is often the most efficient algorithm provided by Encog for supervised feedforward neural networks (Heaton, 2011). Like in the Manhattan Update Rule only the magnitude of the gradient descent is used. Unlike the Manhattan Update rule the RPROP algorithm doesn't use a fixed constant to update the weights. Instead RPROP keeps a delta value for each weight matrix value, which are first initialized to a small value (Heaton, 2011). The magnitude, delta and gradient are used to train each weight matrix individually.
- **Scaled Conjugate Gradient (SCG)** is based on a class of optimization algorithms called Conjugate Gradient Methods (Heaton, 2011). SCG is not applicable for all data sets, but is according to Heaton (2011) quite efficient when applicable (Heaton, 2011).
- **Levenberg Marquardt algorithm (LMA)** takes strengths from Newton's Method and gradient descent algorithms (Heaton, 2011). A hybrid method is created by using a damping factor to merge the two approaches (Heaton, 2011).

4.2.2.2 Building the Emotiv Neural Network

When it comes to building the Artificial Neural Network a class named EmotionalNetwork was created. This chapter explains how this class creates a artificial neural networks, based on the CSV log files, using this class.

4.2.2.2.1 Creating the datasets

Because there are multiple sets of logs there is also a need to create multiple datasets. All the GameState and EmotivState data quantified as numbers makes is simple to use one of the DataSet classes the Encog framework provides, MLDataSet.

```
ArrayList<GameState> gameStates = logparser.readGameStatesFromFile(file);
double XOR_INPUT[][] = new double[gameStates.size()][];
double XOR_IDEAL[][] = new double[gameStates.size()][];
int i = 0;
for(GameState gameState : gameStates) {
        XOR_INPUT[i] = gameState.toArray();
        XOR_IDEAL[i] = gameState.getEmotivState().toArray();
        i++;
        }
        MLDataSet trainingSet = new BasicMLDataSet(XOR_INPUT, XOR_IDEAL);
        return trainingSet;
```

This code gets the path of a log file as a parameter and creates GameStates objects from the log and puts it into files. It then creates two arrays it puts the data from the GameState and the corresponding EmotivState into.

4.2.2.2.1 Creating the networks

Artificial neural networks take a set of parameters. The parameters a network takes dictates the size of each layer, how many weights it should have and overall the structure of the artificial neural network.

```
        private FlatNetwork createNewNetwork() {
                FlatNetwork network = new
        FlatNetwork(sInputNeurons,30,30,sOutputNeurons,false);
                network.randomize();
                return network;
        }
```

When constructing a FlatNetwork it takes the following arguments:

- Neurons in the input layer. This is set to the amount of input the artificial neural network will receive, which in turn is determined by the size of the input data. There is a neuron for each value in the input data, which would mean there are 18 neurons in the EmotivNetwork.
- Neurons in the first hidden layer. In order to have a network with a large set of input neurons and a relatively large set of output neurons there also needs to be many. Through trial and error the number settled on 30.
- Neurons in the second hidden layer. Set at 30.
- Neurons in the output layer. This is set to how many values we want the artificial neural network to output. This is decided by how many values there are in the EmotivState datastructures, resulting in this value being set to 5 for the EmotionLearner.
- The last argument taken is whether the tanh activation function (true), or if the sigmoid activation function is wanted (false). The sigmoid function is used for the EmotionLearner because the sigmoid function forces values to be in the positive range. Because all our desired outputs are values between 0 and 1 using the sigmoid activation function makes sense.

4.2.2.2.3 Training the networks

As discussed previously the Encog framework offer a set of propagation training algorithms to train artificial neural networks. Like stated by Jeff Heaton the Resilient Propagation Training algorithm is often the most effective of the six alternatives (Heaton, 2011). Through trying the different propagation training algorithms available it was found that the RPROP algorithm, combined with the sigmoid activation function, trained the network to the lowest error rate quickest.

```
private FlatNetwork train(MLDataSet trainingset, FlatNetwork network, double error) {
		TrainFlatNetworkResilient train = new TrainFlatNetworkResilient(network, trainingset);
		int epoch = 1;
		do {
			train.iteration();
			System.out.println("Epoch #" + epoch + " Error:" + train.getError());
			epoch++;
		} while(train.getError() > error );
		double e = network.calculateError(trainingset);
		System.out.println("Used "+ epoch + " epochs to train. The network's error      is: " + e);

		return network;
	}
```

The EmotionLearner has access to seven logs, one for each country. To take advantage of this the EmotionLearner trains eight different networks. The EmotionLearner trains a network for each country down to a 0.02 error rate, in addition it trains a general network based on all the CSV-logs available. All the neural networks are serialized and saved to eight different files, one file for each network. Each file represents a different FlatNetwork object.

4.2 The Emotion StateCraft module

The Emotion module for the StateCraft was first created by Carlson and Hellevang in 2010, as discussed in a previous chapter (see section 4.1). The implementation of the Emotion module uses the OCC-model, because, as Carlson and Hellevang argues, it has been implemented and proven as a good choice by previous research (Carlson & Hellevang, 2010). The Emotion module created by Carlson and Hellevang includes an interface for creating new emotions and an EmotionSynthesizer that ties all the emotions together and acts as a module in the Strategic layer of the StateCraft game. Because the emotions implemented in the existing Emotion module are different than the emotions the Emotiv Headset outputs it was decided that the work done in this project would expand, and not replace, the Emotion module already implemented. The main difference from how Carlson and Hellevang (2010) implements their Emotion module is that the extension outlined in this section uses an artificial neural network in order to analyze the GameState to get the emotion intensities.

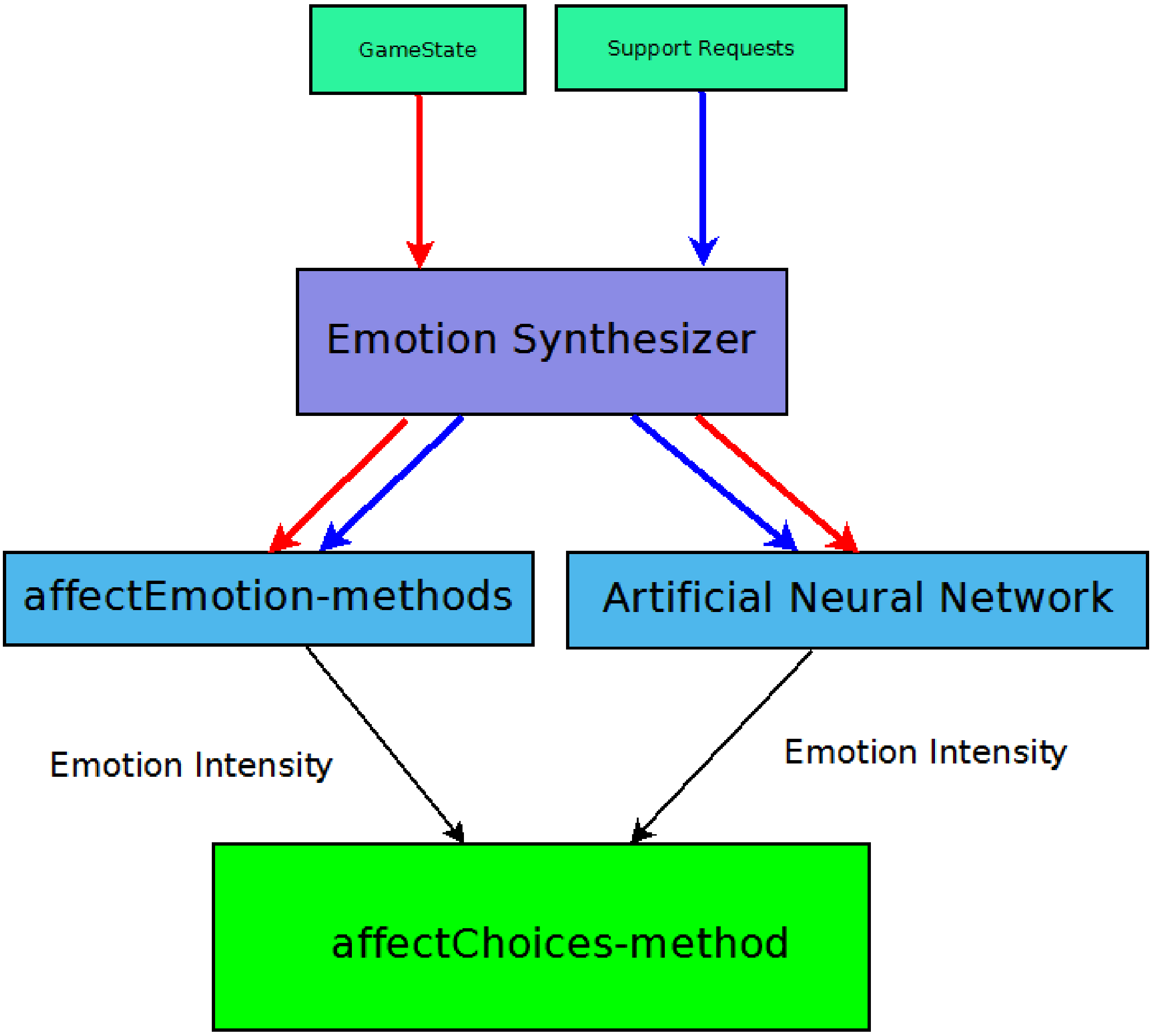


Figure 4.2: Two ways of producing Emotion Intensity in the Emotion Module

As seen in figure 4.2, the EmotionSynthesizer will receive support request messages and new GameStates, and passes them to the appropriate method to compute the emotion intensity for each emotion. Carlson and Hellevang have affectEmotion-methods that dissect and analyze the GameState in order to produce an intensity value for that emotion. This section will outline how the expanded Emotion model uses an artificial neural network to calculate the intensity.

4.2.1 The EmotivModel

is the class responsible for loading and creating the Neural Networks created and imported from the Emotion Learner (see section 4.2). Because of the work done in the Emotion Logger and learner phases of the project there is a neural network for each of the seven countries available. There is also an artificial neural network trained from all the different countries data available, referred to as the general neural network. The EmotivModel loads in a serialized network for the corresponding country, e.g. the AI agent for England will load in the serialized neural network made for England.

GameStates are received through the EmotionSynthesizers receive method. The receive method updates all the emotions based on the new Gamestate, the support requests received and the support requests sent that round. The EmotivModel is sent these data when the new GameStates are received. When this happens the EmotivModel class converts the data into the the format the neural network accepts, as specified in section 4.2.2.2. The output of the neural network is then calculated and saved to make it available to the frustration, excitement and engagement (see following sections).

In order to add emotions to the Emotion Module a class which implements the Emotion interface has to be created and added to the EmotionSynthesizers ArrayList of emotions. The Emotion interface was created by Carlson and Hellevang, and is described in more detail in section 3.3. As described in section 3.3.3.2 the Emotion interface contains the methods affectEmotions() and affectChoices(). The affectEmotions() method will use the artificial neural network through the EmotivModel class to compute the emotional intensity. While affectChoices() will change the orders for the next round.

4.2.2 Frustration

One of the emotions that can be read from the Emotiv Epoc Headset is frustration. A frustrated player will act similarly to an angry player, but frustration is not directed towards an opponent. The affectChoices() method will increase all attack orders with a small random value, as well as lower the chance of giving any opponents support.

4.2.3 Excitement

The excited player will be confident that luck is on his side, and more friendly and giving towards other players. For the Excitement class affectDecision() will increase the factual value of all orders with a magnitude decided by the potential value and the intensity of excitement. In addition it will increase the factual value of all support orders.

4.2.4 Engagement

An engaged player will want to prolong stay in the game for as long as possible by prioritizing orders with high factual values. The affectDecision()-method will boost the factual value of an order by a small amount based on the potential value and the emotion intensity.

4.3 Tools

To keep backups and revisions of the work a private GitHub repository was used, provided to me through a GitHub educational account they were kind enough to supply me with (GitHub, 2012). GitHub makes it possible to split projects into branches and keep track of issues for each branch.

Eclipse is a well known tool for programming in Java. Considering that all the systems are implemented in Java, Eclipse became an obvious choice for an Integrated Development Environment (IDE). Eclipse provides very good debugging for Java, and will also provide a simple way of running and testing Java code through the console and their testing suites.

The goal of the thesis is to use the emotion data from the Emotiv Epoc headset to improve on the existing Emotion module in StateCraft. In order to create an AI system based on data one either needs to analyze that data manually, and implement the agent according to the analytical findings, or use a machine learning algorithm.

Table 5.1 - Simulation Set Up

Configuration	Turkey	Austria	Italy	Germany	Russia	France	England	Amount
All Emo+Emotiv	Emo2	Emo2	Emo2	Emo2	Emo2	Emo2	Emo2	50
All Emotiv	Emotiv	Emotiv	Emotiv	Emotiv	Emotiv	Emotiv	Emotiv	50
All Emotiv-G	Emotiv-G	Emotiv-G	Emotiv-G	Emotiv-G	Emotiv-G	Emotiv-G	Emotiv-G	50
No Emo								50
E201	Emo2							30
E202		Emo2						30
E203			Emo2					30
E204				Emo2				30
E205					Emo2			30
E206						Emo2		30
E207							Emo2	30
E201-G	Emo2-G							30
E202-G		Emo2-G						30
E203-G			Emo2-G					30
E204-G				Emo2-G				30
E205-G					Emo2-G			30
E206-G						Emo2-G		30
E207-G							Emo2-G	30
EMOTIV101	Emotiv							30
EMOTIV102		Emotiv						30
EMOTIV103			Emotiv					30
EMOTIV104				Emotiv				30
EMOTIV105					Emotiv			30
EMOTIV107						Emotiv		30
EMOTIV101-G	Emotiv-G						Emotiv	30
EMOTIV102-G		Emotiv-G						30
EMOTIV103-G			Emotiv-G					30
EMOTIV104-G				Emotiv-G				30
EMOTIV105-G					Emotiv-G			30
EMOTIV106-G						Emotiv-G		30
EMOTIV107-G							Emotiv-G	

Chapter 5

5 Evaluation of the new Emotion module

In this chapter the changes to the Emotion module will be evaluated. In the introduction to the thesis the research questions for the Emotiv trained Emotion module were outlined:

RQ1: How does modeling emotions of players affect agent performance?
RQ1.a: Does an agent perform better with emotions than without emotions?
RQ1.b: Does an agent using country specific emotions perform better than an agent using general emotions?
RQ2: Does an agent trained from EEG-data perform better than an agent that is based on game states?

The research questions will be explored through statistical analysis of data collected through simulations. To analyze the normally distributed data a paired sample t-test is used . The data that is not normally distributed a Wilcoxon singed-rank test is employed (Wohlin, et al., 2000). The t-tests are all set to use a 95% confident interval. To test the data for normality a Shapiro-Wilk test is used (Shapiro & Wilk, 1965).

5.1 Simulations

In order to make the data gathered comparable to the data gathered by Carlson and Hellevang (2010) it was decided to run simulations from 1901 to 1911, where all countries were controlled by agents. Table 5.1 depicts the configurations used for the simulations. The blank cells mean the agent did not use the Emotion module. The cell value Emo2 means the agent used both the emotions from the Emotiv trained artificial neural network and the emotions created by Carlson and Hellevang (2010). The -G notation means the neural network is running the artificial network that is trained based on all the logs. The cell value Emotiv means that the Emotion module is running with the emotiv emotions only.
The E201 through E207 are simulations where each country is running the full Emotion module with their country specific artificial neural network, in E201-G through E207-G configurations the agent is running the general neural network. In the EMOTIV101 through EMOTIV103 the Emotion module consists of only the emotiv emotions, in the EMOTIV101-

G through EMOTIV103-G the general neural network is loaded. The amount column describes how many simulations were ran with each configuration.

From the simulations the Mean, Standard Deviation, Median, number of victories and number of extinctions were calculated for each country. The number of victories is where the country finished with the highest amount of supply centers, if the game resulted in a tie multiple winners were credited. Extinctions is where the country finished the game with zero supply centers left.

Table 5.2 - No Emo Simulation Results

Country	Mean	Std dev	Median	Victories	Extinctions
Germany	5.50	3.454	5	18	0
Turkey	5.78	1.657	6	8	0
England	5.27	2.150	5	10	0
Austria	4.70	2.297	5	4	2
France	4.92	2.127	5	9	0
Italy	4.46	1.908	4	7	1
Russia	3.38	2.311	3	6	3

Table 5.3 - All Emo+Emotiv (Emo2) Simulation Results

Country	Mean	Std dev	Median	Victories	Extinctions
Germany	5.70	2.169	6	15	0
Turkey	5.66	1.912	5	15	0
England	3.80	0.904	4	2	0
Austria	4.14	2.020	4	4	2
France	5.64	1.396	5,5	9	0
Italy	4.20	1.370	4	1	1
Russia	5.58	2.588	5	14	2

Table 5.4 - All Emotiv Simulation Results

Country	Mean	Std dev	Median	Victories	Extinctions
Germany	5.72	1.539	6	16	0
Turkey	5.88	1.674	6	15	0
England	4.16	1.095	4	1	0
Austria	4.52	1.951	4	12	1
France	5.06	1.531	5	8	0
Italy	4.36	1.588	4	5	0
Russia	4.24	1.791	5	7	0

Table 5.5 - All Emotiv-G Simulation Results

Country	Mean	Std dev	Median	Victories	Extinctions
Germany	5.16	2.103	5	17	0
Turkey	6.44	1.656	6,5	23	0
England	3.98	1.134	4	3	0
Austria	4.16	1.899	4	8	0
France	4.94	1.531	5	6	0
Italy	4.40	1.485	4	3	0
Russia	4.78	2.359	5	9	1

Comparing the results from No Emo in Table 5.2 and the results from All Emo + Emotiv in Table 5.3. Russia won more with emotions enabled, while England won less. The Mean and Median numbers for England and Russia are also change The All Emotiv results also show some considerable differences in amount of victories, most notably in Austria with 12 victories versus 4 in both of the other result sets. The standard deviations are closer to each other in the All Emotiv result set. When the agents use the general artificial neural network it seems that the Mean, Standard Deviation and Median values are relatively similar to the Emotiv results.

Table 5.6 - Results from E201 through 207

Country	Mean	Std dev	Median	Victories	Extinctions
Germany	2.67	2.155	2	2	5
Turkey	5.5	1.479	6	6	0
England	4.20	1.243	4	2	0
Austria	3.97	2.205	4	1	1
France	4.07	1.799	4	3	1
Italy	4.5	2.076	5	4	1
Russia	1.93	1.552	2	0	5

Comparing Table 5.2 and 5.6 one can see that the values for England and Germany decrease substantially when compared to their performance with no emotions. Turkey is the agent that performs best with the EMO2 set up.

Table 5.7 - Results from E201-G through 207-G

Country	Mean	Std dev	Median	Victories	Extinctions
Germany	2.67	1.936	3	2	5
Turkey	5.70	2.103	6	8	0
England	3.93	1.388	4	3	0
Austria	4.03	2.282	4	4	1
France	4.00	2.051	4,5	2	3
Italy	4.87	1.502	5	2	0
Russia	1.90	2.310	1	3	8

In Table 5.7 one can see that some small differences in how the countries perform. The most notable difference is England's decrease from 4.20 in the E207 to 3.93 in the E207-G simulation. It is also worth noting that Turkey increase their mean number of supply centers with the general network versus the country specific network in Table 5.6.

Table 5.8 - Results from EMOTIV101 through EMOTIV107

Country	Mean	Std dev	Median	Victories	Extinctions
Germany	5.47	1.871	3	1	0
Turkey	5.60	2.328	5.5	6	0
England	3,73	0.907	4	0	0
Austria	3.63	2.312	4	3	5
France	4.24	1.675	4	1	1
Italy	3.67	1.124	4	0	0
Russia	2.733	1.230	3	0	0

Table 5.9 - Results from EMOTIV101-G through EMOTIV107-G

Country	Mean	Std dev	Median	Victories	Extinctions
Germany	3.10	1.583	3	0	2
Turkey	4.97	1.650	5	5	0
England	3.73	1.048	4	0	0
Austria	3.57	1.794	3	3	1
France	3.467	1.525	4	1	1
Italy	3.53	1.074	3.5	0	0
Russia	2.90	1.882	2.5	1	2

The countries that change the most when running the general network versus the country tailored network is Germany and Turkey.

5.1.1 Results for individual countries

In order to find statistically significant the results from each country will be analyzed with paired-sample tests. The samples used will be the number of supply centers a country has in 30 simulations. In the following sections results from paired-sample tests will be presented while in the following sections a summary and further analysis will be done.

5.1.1.1 Turkey

Table 5.10: Emotion's effect on performance: Turkey

a) Paired Samples Statistics

Configuration	Mean	N	Std. Deviation
No emo	4.80	30	1.095
E201	5.47	30	1.479
E201-G	5.70	30	2.103
EMOTIV101	5.60	30	2.328
EMOTIV101-G	4..97	30	1.650

b) Paired Sample Test: E201 vs No emo

Turkey	Opponents	Mean difference	Z	p-value
Emo2	Regular	-0.67	-1.74	0.0811

c) Paired Sample Test: E201-G vs No emo

Turkey	Opponents	Mean difference	t-value	p-value
Emo2-G	Regular	-0.9	-2.103	0.044

d) Paired Sample Test: EMOTIV101 vs No emo

Turkey	Opponents	Mean difference	t-value	p-value
Emotiv	Regular	-0.8	-1.849	0.075

e) Paired Sample Test: EMOTIV101-G vs No emo

Turkey	Opponents	Mean difference	Z	p-value
Emotiv-G	Regular	-0.17	-1.48	0.1389

Since only one of the samples used in the E201 vs No Emo test was normally distributed a Wilcoxon singed-rank test was conducted to compare the numbers from E201 and No Emo. The Wilcoxon singed-rank test outputs the values $Z = -1.74$ and $p = 0.0811$, as seen in Table 5.11 b). The number of supply centers increased from 4.80 (+/- 1.095) to 5,47 (+/-1.479), showing a very small improvement when Turkey runs with the extended Emotion module. The results in b) are not statistically significant because $p = 0.0811 > 0.05$.

To analyze the results in c) a paired sample t-test was used because both of the data samples were normally distributed. As seen in Table 5.11 c) the t-test gave the numbers $t = -2.103$ and p=0.044. The difference in supply centers shows a very small improvement, going from 4.80(+/- 1.095) to 5.70 (+/- 2,103). The results in c) are statistically significant since p = 0.044 < 0.05.

Both the data samples for No emo and EMOTIV101 are normally distributed so a paired sample t-test is used to analyze the data. The t-test gives the values t=-1.849 and p=0.075. The difference in mean shows a small improvement, going from 4.80 (+/- 1.095) to 5.60 (+/- 2.328). Because p = 0.075 > 0.05 the results in d) are not statistically significant.

Only one of the data samples in e) are normally distributed so a Wilcoxon singed-rank test was conducted. Table 5.11 e) shows the values Z=-1.48 and p = 0.1389. The improvement shown in e) is smaller than the other results for Turkey. With the EMOTIV-G set up Turkey gets a mean of 4.97 (+/- 1.650), versus 4.80 (+/- 1.095) with no emotions. Because p = 0.1389 < 0.05 the results are no statistically significant.

5.1.1.2 Austria

Table 5.11: Emotion's effect on performance: Austria

a) Paired Samples Statistics

Configuration	Mean	N	Std. Deviation
No emo	4.40	30	2.500
E202	3.97	30	2.205
E202-G	4.03	30	2.282
EMOTIV102	3.63	30	2.312
EMOTIV102-G	3.57	30	1.794

b) Paired Sample Test: E202 vs No emo

Austria	Opponents	Mean difference	t-value	p-value
Emo2	Regular	0.43	0.663	0.512

c) Paired Sample Test: E202-G vs No emo

Austria	Opponents	Mean difference	t-value	p-value
Emo2-G	Regular	0.37	0.594	0.557

d) Paired Sample Test: EMOTIV102 vs No emo

Austria	Opponents	Mean difference	t-value	p-value
Emotiv	Regular	0.77	1.167	0.253

e) Paired Sample Test: EMOTIV102-G vs No emo

Austria	Opponents	Mean difference	t-value	p-value
Emotiv-G	Regular	0.83	1.205	0.238

A paired-sample t-test was used in Table b) because both data samples were normally distributed. The t-test gave the values $t = 0.663$ and $p = 0.512$. Austria performs slightly worse with the E202 setup, going from 4.40 (+/- 2.500) to 3.97 (+/- 2.205). The results are not statistically significant since $p = 0.512 > 0.05$.

The samples in Table c) are both normally distributed so a paired-sample t-test was used. The t-test gave the values $t = 0.594$ and $p = 0.557$. Austria performs slightly worse with the E202-G setup, going from 4.40 (+/- 2.500) to 4,03 (+/- 2.282). The results are not statistically significant since $p = 0. 557 > 0.05$.

In the EMOTIV102 vs No emo analysis seen in Table 5.11 d) a paired-sample t-test was used because both samples were normally distributed. The analysis give the numbers $t = 1.167$ and $p = 0.253$. Austria performs worse in d) than the simulations in b) and c), going from 4.40 (+/- 2.500) to 3.63 (+/- 2.312). Because $p = 0.253 > 0.05$ the results are not statistically significant.

The data samples in e) were also normally distributed, resulting in a paired-sample t-test being used in the comparison. The t-test gave the results $t =1.205$ and $p = 0.238$. With the EMOTIV102-G setup Austria performs the worst of the four Emotion set ups used. Austria goes from 4.40 (+/- 2.500) to 3.57 (+/- 1.794). Since $p = 0.253 > 0.05$ the results are not statistically significant.

5.1.1.3 Italy

Table 5.12: Emotion's effect on performance: Italy

a) Paired Samples Statistics

Configuration	Mean	N	Std. Deviation
No emo	4.60	30	1.812
E203	4.37	30	2.076
E203-G	4.87	30	1.502
EMOTIV103	3.67	30	1.124
EMOTIV103-G	3.53	30	1.074

b) Paired Sample Test: E203 vs No emo

Italy	Opponents	Mean difference	t-value	p-value
Emo2	Regular	0.23	0.053	0.958

c) Paired Sample Test: E203-G vs No emo

Italy	Opponents	Mean difference	t-value	p-value
Emo2-G	Regular	-0.27	-0.861	0.396

d) Paired Sample Test: EMOTIV103 vs No emo

Italy	Opponents	Mean difference	t-value	p-value
Emotiv	Regular	0.93	2.948	0.006

e) Paired Sample Test: EMOTIV103-G vs No emo

Italy	Opponents	Mean difference	Z	p-value
Emotiv-G	Regular	1.07	2.30	0.0215

The data samples in for both E203 and No Emo for Italy were normally distributed and a paired-sample t-test was conducted. Italy performed slightly worse with the E203 setup, going from 4.60 (+/- 1.812) to 4.37 (+/- 2.076). The data is not statistically significant since $p = 0.958 > 0.05$.

A paired-sample t-test was conducted in c) as well because both data samples were normally distributed. The t-test showed the numbers $t = -0.861$ and $p = 0.396$. With the E203-G setup Italy improved slightly over the No emo set up, going from 4.60 (+/- 1.812) to 4.87 (+/- 1.502). The data is not statistically significant since p = 0. 396> 0.05.

The data samples in d) were also normally distributed, resulting in a paired-sample t-test being used in the comparison. The t-test gave the results t =1.205 and p = 0.238. Italy goes from 4.60 (+/- 1.812) to 3.67 (+/- 1.124). Since p = 0. 006> 0.05 the results are statistically significant.

Only one of the data samples in e) are normally distributed so a Wilcoxon singed-rank test was conducted. Table 5.12 e) shows the values Z=-1.48 and p = 0.1389. The improvement shown in e) is smaller than the other results for Turkey. With the EMOTIV-G set up Italy goes from 4.60(+/- 1.812) with no emotions to 3.53 (+/- 1.074) with the EMOTIV-G emotional setup. Because p = 0. 0215< 0.05 the results are statistically significant.

5.1.1.4 Germany

Table 5.13: Emotion's effect on performance: Germany

a) Paired Samples Statistics

Configuration	Mean	N	Std. Deviation
No emo	5.7	30	1.539
E204	2.67	30	2.155
E204-G	2.67	30	1.936
EMOTIV104	5.5	30	1.871
EMOTIV104-G	3.1	30	1.583

b) Paired Sample Test: E204 vs No emo

Germany	Opponents	Mean difference	Z	p-value
Emo2	Regular	3.03	3.18	0.0015

c) Paired Sample Test: E204-G vs No emo

Germany	Opponents	Mean difference	Z	p-value
Emo2-G	Regular	3.03	3.36	0.0008

d) Paired Sample Test: EMOTIV104 vs No emo

Germany	Opponents	Mean difference	Z	p-value
Emotiv	Regular	0.2	2.75	0.0059

e) Paired Sample Test: EMOTIV104-G vs No emo

Germany	Opponents	Mean difference	Z	p-value
Emotiv-G	Regular	2.6	3.45	0.0006

None of the samples in b) were normally distributed so a Wilcoxon singed-rank test was done. The Wilcoxon test gave the numbers Z = 3.18 and p = 0.0015. Germany performed significantly worse with the E204 set up, going from 5.70 (+/- 1.539) to 2.67 (+/-2.155). The data is statistically significant because p = 0.0015 < 0.05.

Because none of the samples in c) were normally distributed a Wilcoxon singed-rank test was done. The analysis showed the number Z = 3.36 and p = 0.0008. Germany performed significantly worse with the E204-G set up, producing the same mean as the E204 set up. The data is statistically significant because p = 0.0008 < 0.05.

From the EMOTIV104 simulations none of the data samples were normally distributed, so a Wilcoxon singed-rank test was used. The Wilcoxon test produced the numbers Z = 2.75 and p = 0.0059. With the EMOTIV104 set up Germany performs only slightly worse, going from 5.70 (+/- 1.539) to 5.5 (+/-1.871). The data is statistically significant because p = 0.0059 < 0.05.

A Wilcoxon singed-rank test was used in Table 5.13 e) because only one of the data samples was normally distributed. The numbers from the Wilcoxon test are: Z = 3.45 and p = 0.0006. Germany performs significantly worse in with the general network, the mean number of supply centers went from 5.70 (+/- 1.539) to 53.1 (+/-1.583). The data is statistically significant because p = 0.0006 < 0.05.

5.1.1.5 Russia

Table 5.14: Emotion's effect on performance: Russia

a) Paired Samples Statistics

Configuration	Mean	N	Std. Deviation
No emo	3.60	30	2.311
E205	1.93	30	1.552
E205-G	1.90	30	2.310
EMOTIV105	2.73	30	1.230
EMOTIV105-G	2.9	30	1.882

b) Paired Sample Test: E205 vs No emo

Russia	Opponents	Mean difference	Z	p-value
Emo2	Regular	1.67	2.47	0.0134

c) Paired Sample Test: E205-G vs No emo

Russia	Opponents	Mean difference	Z	p-value
Emo2-G	Regular	1.7	2.71	0.0066

d) Paired Sample Test: EMOTIV105 vs No emo

Russia	Opponents	Mean difference	Z	p-value
Emotiv	Regular	0.87	1.77	0.0774

e) Paired Sample Test: EMOTIV105-G vs No emo

Russia	Opponents	Mean difference	Z	p-value
Emotiv-G	Regular	0.7	1.17	0.2412

Only one of the samples in b) was normally distributed so a Wilcoxon singed-rank test was conducted. The Wilcoxon test gave the numbers Z = 2.47 and p = 0.0134. Russia performs worse with the Emo2 set up, going from 3.60 (+/- 2.311) with no emotions to 1.93 (+/- 1.552). The results are statistically significant since p = 0.0134 < 0.05.
A Wilcoxon singed-rank test was also used in c) since only one of the samples was normally distributed. From the test the numbers Z = 2.71 and p = 0.0066 were found. Russia performs slightly worse than E205 seen in b) when using the E205-G set up. The number of supply

centers goes from 3.60 (+/- 2.311) to 1.90 (+/- 2.310). The results are statistically significant since p = 0.0066 < 0.05.

From the EMOTIV105 simulations (seen in d)) only one of the data samples was normally distributed, so a Wilcoxon singed-rank test was used. As seen in d) the results from the Wilcoxon test are: Z = 1.77 and p = 0.0774. Russia performs better with the EMOTIV setup than with the E205 and E205-G set ups. The number of supply centers goes down from 3.60 (+/- 2.311) to 2.73 (+/- 1.230). However the results are not statistically insignificant since p = 0.0774 > 0.05.

In the last simulation for Russia none of the data samples were normally distributed so a Wilcoxon singed-rank test was conducted. The test resulted in the numbers Z = 1.17 and p = 0.2412. Russia's number of supply centers went from 3.60 (+/- 2.311) to 2.9 (+/- 1.882). The results are not statistically significant because p = 0.2412 > 0.05.

5.1.1.6 France

Table 5.15: Emotion's effect on performance: France

a) Paired Samples Statistics

Configuration	Mean	N	Std. Deviation
No emo	4.67	30	2.127
E206	4.07	30	1.799
E206-G	4.00	30	2.051
EMOTIV106	4.24	30	1.675
EMOTIV106-G	3.47	30	1.525

b) Paired Sample Test: E206 vs No emo

France	Opponents	Mean difference	Z	p-value
Emo2	Regular	0.6	1.87	0.0617

c) Paired Sample Test: E206-G vs No emo

France	Opponents	Mean difference	Z	p-value
Emo2-G	Regular	0.67	1.57	0.1164

d) Paired Sample Test: EMOTIV106 vs No emo

France	Opponents	Mean difference	Z	p-value
Emotiv	Regular	0.24	0.73	0.4662

e) Paired Sample Test: EMOTIV106-G vs No emo

France	Opponents	Mean difference	Z	p-value
Emotiv-G	Regular	1.2	2.48	0.0132

In the comparison done in b) none of the data samples were normally distributed, so a Wilcoxon singed-rank test was conducted. The test gave the numbers Z = 1.87 and p = 0.0617. France performs somewhat worse with the E206, going from 4.67 (+/- 2.127) to 4.07 (+/- 1.799). Because $p = 0.0617 > 0.05$ the results are not statistically significant.
Because only one of the data samples in c) follow normal distribution a Wilcoxon singed-rank test Is used to produce the numbers Z = 1.57 and p = 0.1164. France goes from 4.67 (+/- 2.127) to 4.00 (+/- 2.051). The results are not statistically significant since $p = 0.1164 > 0.05$.
In the comparison done in b) none of the data samples were normally distributed, so a Wilcoxon singed-rank test was conducted. The test gave the numbers Z = 0.73 and p = 0.4662. France only performs slightly worse with the Emotiv set up, going from 4.67 (+/- 2.127) to 4.24 (+/- 1.675). However the results are not statistically significant because $p = 0.4662 > 0.05$.
In the comparison between EMOTIV106-G and No emo for France, only one of the data samples was normally distributed resulting in a Wilcoxon singed-rank test being used. The Wilcoxon test produced the numbers Z = 2.48 and p = 0.0132. France performs the worst of all b), c), d) and e) with the EMOTIV106-G set up. The number of supply centers goes from 4.67 (+/- 2.127) to 3.47 (+/- 1.525). The results are statistically significant because $p = 0.0132 < 0.05$.

5.1.1.7 England

Table 5.16: Emotion's effect on performance: England

a) Paired Samples Statistics

Configuration	Mean	N	Std. Deviation
No emo	5.27	30	0.904
E207	4.2	30	1.243
E207-G	3.93	30	1.388
EMOTIV107	3.73	30	0.907
EMOTIV107-G	3.73	30	1.048

b) Paired Sample Test: E207 vs No emo

England	Opponents	Mean difference	Z	p-value
Emo2	Regular	1.07	1.85	0.0645

c) Paired Sample Test: E207-G vs No emo

England	Opponents	Mean difference	Z	p-value
Emo2-G	Regular	1.34	2.12	0.0340

d) Paired Sample Test: EMOTIV107 vs No emo

England	Opponents	Mean difference	Z	p-value
Emotiv	Regular	1.54	2.71	0.0067

e) Paired Sample Test: EMOTIV107-G vs No emo

England	Opponents	Mean difference	Z	p-value
Emotiv-G	Regular	1.54	3.07	0.0021

Only one of the samples in the paired sample test of E207 and England's no emo results were normally distributed, and a Wilcoxon singed-rank test was used. The results of the test, seen in Table 5.16, were $Z = 1.85$ and $p = 0.0645$. England performs worse with the E207 setup, going from 5.27 (+/- 0.904) to 4.2 (+/- 1.243). The results are not statistically significant because $p = 0.0645 > 0.05$.

None of the samples in c) were normally distributed so a Wilcoxon singed-rank test was used. The test results were as follows: $Z = 2.12$ and $p = 0.0340$. England performs worse with the general network compared to the country specific network. The number of supply centers decreases from 5.27 (+/- 0.904) to 3.93 (+/- 1.388). Because $p = 0.0340 < 0.050$ the results are statistically significant.

Only one of the samples in d) were normally distributed so a Wilcoxon singed-rank test was used. The test gave the results: $Z = 2.12$ and $p = 0.0340$. The number of supply centers decreases from 5.27 (+/- 0.904) to 3.73 (+/- 0.907) using the EMOTIV set up. Because $p = 0.0067 < 0.050$ the results are statistically significant.

A Wilcoxon singed-rank test was used in e) because only one of the samples was normally distributed. The test results were: $Z = 3.07$ and $p = 0.0021$. England performs the same with the general network compared to the country specific network when using the EMOTIV emotion set up. The number of supply centers decreased from 5.27 (+/- 0.904) to 3.73 (+/- 1.048). Because p = 0.0.0021 < 0.050 the results are statistically significant.

5.1.2 Summary

Table 5.17: Summary of the Carlson's E101 trough E107 analysis

Country	Config 1	Config 2	Mean Difference	Improvement	Test	Significance
Turkey	No Emo	E101	-0.033	0.63%	Wilcoxon	0.954
Austria	No Emo	E102	0.267	-5.8%	Paired t-test	0.627
Italy	No Emo	E103	-0.433	10.6%	Wilcoxon	0.279
Germany	No Emo	E104	2.267	-32.2%	Paired t-test	0.017
Russia	No Emo	E105	0.967	-32.6%	Wilcoxon	0.155
France	No Emo	E106	-0.700	16.7%	Paired t-test	0.373
England	No Emo	E107	0.100	-2.1%	Wilcoxon	0.897
All	**No Emo**	**E101...E107**	**0.348**	**-7.2%**	**Wilcoxon**	**0.276**

Table 5.17 presents the summary of the statistical analysis done by Carlson and Hellevang (2010), with their own Emotion module (Carlson & Hellevang, 2010). Only the results from the Germany (E104) simulations are statistically significant. Germany has a big decrease in performance, with 2.267 supply centers less compared to the No emo ran by Carlson and Hellevang.

Table 5.18: Summary of the EMO2 simulations

Country	Config 1	Config 2	Mean Difference	Improvement	Test	Significance
Turkey	No Emo	E201	-0.67	12.24%	Wilcoxon	0.0811
Austria	No Emo	E202	0.43	-9.77%	Paired t-test	0.512
Italy	No Emo	E203	0.23	-5%	Paired t-test	0.958
Germany	No Emo	E204	3.03	-53.16%	Wilcoxon	0.0015
Russia	No Emo	E205	1.67	-46.39%	Wilcoxon	0.0134
France	No Emo	E206	0.6	-12.85%	Wilcoxon	0.0617
England	No Emo	E207	1.07	-20.3%	Wilcoxon	0.0645
All	**No Emo**	**E201...E207**	**0.909**	**-19.26%**	**Wilcoxon**	**0.0003**

As seen in Table 5.18 the EMO2 simulations give worse results than the EMO results presented by Carlson and Hellevang. Looking at the results across all the simulations it looks like the EMO2 simulations perform overall worse, going from a -7.2% 'improvement' compared to no emotions versus -19.26% 'improvement'. However, the summary of Hellevang and Carlsons simulations are not statistically significant. In order to answer RQ2 individual differences will have to be analyzed (see chapter 5.1.4).

Table 5.19: Summary of the EMO2-G simulations

Country	Config 1	Config 2	Mean Difference	Improvement	Test	Significance
Turkey	No Emo	E201-G	-0.9	15.79%	Paired t-test	0.044
Austria	No Emo	E202-G	0.37	-8.41%	Paired t-test	0.557
Italy	No Emo	E203-G	-0.27	5.54%	Paired t-test	0.396
Germany	No Emo	E204-G	3.03	-52.16%	Wilcoxon	0.0008
Russia	No Emo	E205-G	1.7	-47.22%	Wilcoxon	0.0066
France	No Emo	E206-G	0.67	-14.35%	Wilcoxon	0.1164
England	No Emo	E207-G	1.34	-25.42%	Wilcoxon	0.0340
All	**No Emo**	**E201-G...E207-G**	**0.848**	**-17.97%**	**Wilcoxon**	**0.0006**

Table 5.19 shows that the EMO2-G, with a -17.47% improvement, performs slightly better than the EMO2 simulation, which has a -19.26% improvement compared to no emotions.

These results indicate a negative answer to RQ1.b, but because this is a mix of emotions it is not a definitive answer. RQ1.b will be discussed further in the rest of this section.

Table 5.20: Summary of the EMOTIV simulations

Country	Config 1	Config 2	Mean Difference	Improvement	Test	Significance
Turkey	No Emo	EMOTIV101	-0.8	14.29%	Paired t-test	0.075
Austria	No Emo	EMOTIV102	0.77	-17.50%	Paired t-test	0.253
Italy	No Emo	EMOTIV103	0.93	-20.22%	Paired t-test	0.006
Germany	No Emo	EMOTIV104	0.2	-3.51%	Wilcoxon	0.0059
Russia	No Emo	EMOTIV105	0.87	-24.17%	Wilcoxon	0.0774
France	No Emo	EMOTIV106	0.24	-9.21%	Wilcoxon	0.4662
England	No Emo	EMOTIV107	1.54	-29.22%	Wilcoxon	0.0067
All	**No Emo**	**EMOTIV101... EMOTIV107**	**0.890**	**-18.86%**	**Wilcoxon**	**0.0001**

Table 5.20 shows a summary of the EMOTIV simulations. The EMOTIV performs slightly better than the EMO2 simulations, but slightly worse than the EMO2-G simulations. Overall the performance of the EMOTIV gets an 18.86% decrease in performance compared to no emotions. The EMOTIV set up performs very slightly better compared to EMO2, but slightly worse compared to EMO2-G.

Table 5.21: Summary of the EMOTIV-G simulations

Country	Config 1	Config 2	Mean Difference	Improvement	Test	Significance
Turkey	No Emo	EMOTIV101-G	-0.17	3.42%	Wilcoxon	0.1389
Austria	No Emo	EMOTIV102-G	0.83	-18.86%	Paired t-test	0.238
Italy	No Emo	EMOTIV103-G	1.07	-23.26%	Wilcoxon	0.0215
Germany	No Emo	EMOTIV104-G	2.6	-45.61%	Wilcoxon	0.0006
Russia	No Emo	EMOTIV105-G	0.7	-19.44%	Wilcoxon	0.2412
France	No Emo	EMOTIV106-G	1.2	-25.70%	Wilcoxon	0.0132
England	No Emo	EMOTIV107-G	1.54	-29.22%	Wilcoxon	0.0021
All	**No Emo**	**EMOTIV101-G... EMOTIV107-G**	**1.109**	**-23.5%**	**Wilcoxon**	**0.0001**

The results from EMOTIV-G, in Table 5.21, presents by far the worst results of the 4 emotion set ups. The mean difference in EMOTIV was at 0.890 (-18.86% improvement), while with the general network the difference across all simulations was at 1.109 (-23.5% improvement). These results would indicate that agents perform worse when using a general versus using a country specific neural network for emotions, which gives a negative answer to RQ1.b.

5.1.3 Emotiv emotion occurrences

In order to see if the Emotiv emotions influence the decision making the occurrences of each emotion was counted. An emotion occurs if the intensity is larger than 50. This means multiple emotions can be counted at the same time. The occurrences were counted for the EMOTIV and EMOTIV-G set ups, over every GameState entry in the simulation logs.

Table 5.22: Occurrences of emotions in the EMOTIV simulations

Country	Frustration	Excitement	Engagement	# of GameStates
Turkey	368	719	1153	1230
Austria	1067	245	1209	1230
Italy	49	117	1230	1230
Germany	16	25	1230	1230
Russia	863	483	1230	1230
France	952	535	1229	1230
England	67	313	1081	1230
Total	**3382**	**1955**	**8362**	**8610**

Presented in Table 5.22 is the amount of times each emotion was "felt" by the agent during the EMOTIV simulations. Note how Austria, Russia and France has very higher numbers of frustration. Germany, England and Italy has very low frustration, while Turkey has a bit more. Turkey gets the highest number of excitement, while Germany experiences excitement very rarely.

Table 5.23: Occurrences of emotions in the EMOTIV-G simulations

Country	Frustration	Excitement	Engagement	# of GameStates
Turkey	1067	245	1209	1230
Austria	1090	302	1229	1230
Italy	1058	98	1205	1230
Germany	932	359	1215	1230
Russia	801	246	1119	1230
France	996	242	1229	1230
England	1003	78	1223	1230
Total	**6947**	**1570**	**8429**	**8610**

In Table 5.23 the emotion occurrences for the EMOTIV set up using the general network are counted, EMOTIV-G. The most notable difference is that the agents experience frustration about twice as often when using the general network, going from 3382 to 6947 occurrences.

5.1.4 Findings

An answer to each research question has been sought trough the analysis presented above. In this chapter we will discuss what we have found through analyzing the simulations.

5.1.4.1 Findings for all

Tables 5.17, 5.18, 5.19, 5.20 and 5.21 summarize the five different configurations of the Emotion module we have available. Carlson and Hellevang's Emotion module did not produce statistically significant results when looking at all simulations, with a p value of 0.276. Individual differences need to be compared in order to answer if there is a difference in performance between the EMOTIV emotions and the game state based emotions.

From the EMO2, EMO2-G, EMOTIV, and EMOTIV-G simulations the results are statistically significant. EMO2 had a 0.909 mean difference overall compared to no emotions, giving a 19.26% decrease in performance (Table 5.18). For comparison EMO2-G had a 0.848 mean difference compared to no emotions, resulting in a 17.97% decrease in performance (Table 5.19). Showing that the combination of Carlson and Hellevang's emotions with the EMOTIV emotions benefits slightly from having the general network over the country specific one.

The EMOTIV simulations is where the countries have the Emotiv Epoc emotions only, as shown in Table 5.1. The EMOTIV101 through EMOTIV107 simulations show a 0.890 decrease in performance compared to no emotions, giving a 18.86% decrease in performance (Tble 5.20). The performance is very similar to the EMO2 and EMO2-G simulations. EMOTIV has slightly lower performance compared to EMO2-G, and very slightly higher than EMO2. Looking at the EMOTIV-G simulations there are some differences to note. The overall performance of the EMOTIV-G results in a 1.109 mean difference compared to no emotions, resulting in a 23.5% decrease in performance (Table 5.21). This means that the EMOTIV-G shows the worst performance of the four simulations. The reason for this decrease can be seen when comparing Table 5.22 and Table 5.23, where the general network agent experiences frustration about twice as often as the regular agents. For some countries the frustration emotion can have a severe impact on performance as discussed in the following sections.

5.1.4.2 Findings for Turkey

Turkey is the only country which improves performance in all the five simulations. However only the E201-G results are statistically significant, with a p value of 0.044 (Table 5.19). In the E201-G simulation Turkey displays a mean difference of -0.9 supply centers, resulting in a 15.79% increase in performance. The increase in performance Turkey shows when using the general network Emotiv emotions combined with Carlson and Hellevang's emotions can be a result of the high amount of frustration this network outputs. As seen in Table 5.23 the general network has a clear tendency to make the agent experience frustration very often. As Carlson and Hellevang pointed out in their evaluation it is rather safe for Turkey to act aggressively because Turkey can always fall back to its safe starting position (Carlson & Hellevang, 2010).

The EMOTIV101 and EMOTIV101-G simulations may imply that there can be cases where Turkey can act too aggressively. If Turkey acts too aggressive due to frustration and expands too fast, Turkey may start losing supply centers. One can also note that the excitement emotion is much more pronounced in the EMOTIV simulations with 719 occurrences, versus only 245 occurrences in the EMOTIV-G simulations (Table 5.22 and 5.23). It is possible that Turkey benefits from risky fast expansion, but without attacking the enemy too much.

5.1.4.3 Findings for Austria

From the five Tables 5.17 through 5.21 there are no statistically significant data for Austria. It is however interesting to see that Austria has the country specific network that produces the highest amount of frustration occurrences (Table 5.22 and 5.23). The high frequency of frustration can have an effect on the performance because the EMOTIV102 and EMOTIV102-G have a similar decrease in performance compared to no emotions (a 17.56% and a 18.86% decrease respectively), and also a similar amount of frustration occurrences (1067 and 1090 respectively). Austria seems to get punished for being too aggressive as a result of the frustration emotion.

5.1.4.4 Findings for Italy

Because of Italy's easily defendable starting position Carlson and Hellevang (2010) argues that it makes sense for Italy to be able to make riskier moves and get away with it. The results from the E103 simulations present a -0.433 difference in mean supply centers compared to no

emotions, a 10.6% increase in performance (Table 5.17). The E203 and E203-G simulation results would imply that the addition of Emotiv emotions to Italy's emotional specter decreases performance Italy's performance (Table 5.18 and 5.18). As seen in Table 5.17, 5.18 and 5.19 the results from E103, E203, and E203-G are not statistically significant.
The results from the EMOTIV103 and EMOTIV103-G simulations presented in Table 5.20 and 5.21 are statistically significant. EMOTIV103 shows that Italy is affected negatively by the Emotiv Epoc emotions, with a 20.22% decrease in performance (Table 5.20). However, the EMOTIV103-G simulations show a bigger negative influence with a 23.26% decrease in performance (Table 5.21).

5.1.4.5 Findings for Germany

Germany is the only country for which we have statistically significant data from all the five simulations. Germany in Carlson and Hellevang's simulations show a 32.2% decrease in performance from a mean difference of 2.267 compared to no emotions (Table 5.17). This drop in performance is attributed to the emotions Joy and Anger which leads Germany to perform riskful moves (Carlson & Hellevang, 2010). Because of Germanys geographical location risky moves often leads Germany to lose its home provinces (Carlson & Hellevang, 2010). The same tendency can be seen in the E201 and E201-G simulations where Germany gets a 53.16% drop in performance in both simulations (Table 5.18 and 5.19). The additional risky moves done by Germany as a result of the frustration and excitement emotions produces a further drop in performance.
In the EMOTIV104 simulations the reduction in performance is only at 3.51%, with a mean difference of 0.2 compared to the no emotions simulation (Table 5.20). The low drop in performance can be attributed to the low amount of frustration and excitement occurrences the agent experiences when using the country specific network for Germany (Table 5.22). The opposite can be seen in the EMOTIV104-G results. When the agent for Germany uses the general network the agent experiences frustration 932 times and excitement 359 times (Table 5.23). This results in a performance drop of 45.61% compared to no emotion (2.6 mean difference). The results indicate that Germany performs best with a defensive strategy, as first suggested by Carlson and Hellevang (Carlson & Hellevang, 2010).

5.1.4.6 Findings for Russia

Carlson and Hellevang's (2010) E105 simulations that shows a 32.6% decrease in performance are not statistically significant (Table 5.17). Carlson and Hellevang are however able to argue that Russia, with its long and hard to defend border, benefits from defensive emotions over riskier emotions (Carlson & Hellevang, 2010).

The E205 and E205-G produce statistically significant result. The E205 simulations show a 45.49% reduction in performance, versus the 47.22% reduction shown by E205-G (Table 5.18 and 5.19).

Neither of the EMOTIV simulations are statistically significant, but the results may show something useful nonetheless. The EMOTIV105 simulation produces a 24.17% reduction in performance, with frustration occurring 863 times and excitement occurring 483 times (Table 5.20 and 5.22). On the other hand EMOTIV105-G produces a 19.44%, with frustration occurring 801 times and excitement occurring 246 times (Table 5.21 and 5.23). These numbers supports the argument, originally made by Carlson and Hellevang, that riskful emotions like frustration and excitement impact Russia's performance negatively.

5.1.4.7 Findings for France

The only statistically significant results we have access to, in regards to Frances performance with emotions, are from the EMOTIV106-G simulations. In the simulations analyzed by Carlson and Hellevang (2010) France presented a 16.7% improvement in performance compared to no emotions (Table 5.17). Carlson and Hellevang argues that France can benefit from risky moves because France home provinces are easily defendable. The results from E206 and E206-G show drops in performance by 12.84% and 14.35% respectively (Table 5.18 and 5.19).

EMOTIV106 presents a performance decrease as low as 9.21%, while EMOTIV106-G shows a 25.7% decrease (Table 5.20 and 5.21). While the amount of frustration occurrences are similar, at 956 versus 996, the agent using the general network experiences excitement almost half as often (Table 5.22 and 5.23). Since the excitement emotion promotes risky behavior, these results are in line with what one could expect given the arguments presented by Carlson and Hellevang (2010).

5.1.4.8 Findings for England

The simulations ran for England produce significant results in the E207-G, EMOTIV107 and EMOTIV107-G simulations (Table 5.19, 5.20 and 5.21). In the E107 simulation ran by Carlson and Hellevang England presents a 2.1% decrease in performance, while in the E207 simulation the decrease is as high as 20.3% (Table 5.17 and 5.18). The E207-G simulation presents a 25.42% decrease in performance (Table 5.19).

The EMOTIV107 and EMOTIV107-G simulations present some interesting results, by both having a decrease in performance of 29.22%. Quite a significant decrease, and the general network performs exactly the same as the country specific network. When looking at the emotions the two agents experience one can see that the country specific agent experiences frustration 67 times, excitement 313 times and engagement 1081 times. The general network agent experience frustration 1003 times, excitement 78 times and engagement 1223 times. These numbers may be explained by looking at England's geographical location. Because England is on an isolated island it will not be able to prioritize attack orders until later in the game, reducing the possible negative effect of the frustration emotion. Also note that the country specific network experiences engagement considerably more seldom than the general network, meaning it will not boost the factual value of tactics as often.

5.2 Summary

From the simulations we can see that the agents perform worse by modeling players' emotions. The significant results show that the EMO2, EMO2-G, EMOTIV, and EMOTIV-G perform worse than no emotions. The only country performing better in the simulations is Turkey. The suggested reason for the performance increase is that Turkey benefits from risky behavior over being too defensive. Looking at the overall results the agents perform worse with emotions than they do without (Table 5.17, 5.18, 5.19, 5.20, and 5.21).

Comparing the results from the agents using country specific emotions with agents using general emotions there are some differences to note. The EMO2-G simulations perform slightly better than the counterpart EMO2 (Table 5.18 and 5.19). With the EMOTIV and EMOTIV-G results the opposite is true, here the agents using general emotions perform worse than the agent with country specific emotions (Table 5.20 and 5.21). The EMOTIV-G agents are the ones with overall worst performance of the five different emotional agents. The agents using the general network experienced frustration very often, in 6947 gamestates out of 8610 overall (Table 5.23). Frustration leads to aggressive behavior, which can be very punishing for most countries.

Because of the lack of statistically significant data from the EMO simulations done by Carlson and Hellevang it is difficult to compare the results. The data available gives conflicting results. Germany's performance with EEG-data trained emotions is considerably better than with the gamestate based emotions (Table 5.17 and 5.20). An agent trained from EEG-data will in some cases perform better than an agent based on game states.

Chapter 6

6 Conclusion and Future works

6.1 Conclusion

The objective of this thesis was to study whether the performance of the autonomous agents in StateCraft improves when players' affect is modeled and incorporated into the agent's Emotion module. We also wished to see how a country specific model of emotion would compare to a general model of emotion in terms of performance.

To investigate this an Emotion Logger was developed for the StateCraft game. The Emotion Logger couples emotion data from the Emotiv Epoc headset with corresponding StateCraft game states. An Emotion Learner was developed based on artificial neural networks to use the data from the Emotion Learner to model players' emotions. Two models were made; a general model trained from all the logs and a country specific model for each country. The resulting model was then incorporated into the Emotion module in StateCraft. Because of the two different modules four different configurations of the Emotion module were developed. The different configurations were:

- The emotions from the Emotion Leaner, using the general emotion model
- The emotions from the Emotion Leaner, using the country specific emotion model
- The emotions from the Emotion Learner combined with the game state based emotions, using the general emotion model
- The emotions from the Emotion Learner combined with the game state based emotions, using the country specific emotion model

Evaluation of the Emotion module was done in the form of simulations. Thirty games were simulated for each configuration and the performance in terms of supply centers was analyzed.

6.1.1 Design and Development

The development in this thesis was done in three parts, in this section each part of the development will be reflected on.

6.1.1.1 Emotion Logger

The Emotion Logger is an extension of the StateCraft game that couples emotion data from the Emotiv Epoc headset with corresponding StateCraft game states. The implementation of the Emotion Logger attempts to capture all the data about game states relevant to the players' emotions, as well as data representing the player's emotions. The data given by the Emotion Logger affects the implementation and performance of the remaining two parts of the project. Implementing the Emotiv Epoc logging into StateCraft also seemed to affect StateCraft's performance and stability negatively. The decreased performance and stability could have impacted the user experience which in turn could have impacted the emotion readings taken during game play.

The Emotion Logger could benefit from improvements in two areas:

1. Increased performance and stability during game play
2. More detailed data saved about both emotions and game state

6.1.1.2 Emotion Learner

The implementation of the Emotion Learner is affected to a large degree by the work done in the Emotion Logger. The Emotiv Epoc headset gives data about emotions, but not about whom the emotions are directed. With more game state data on whom causes the player trouble or accord it could have been possible to train the artificial neural network additional directed and undirected emotions. By also using the facial expression data from the Emotiv Epoc the Emotion Learner could have offered the possibility of more emotions.

When creating artificial neural networks there are many variables to tweak and fine tune. The structure of the input data and the structure of the desired output dictated how some of these variables were tuned. Other variables were either left with the default value or briefly tuned and tested.

Normalizing the emotion intensities used as output in the network could be a useful improvement as well. The emotions developed by Carlson and Hellevang, with some exceptions, had an emotional specter from zero to one hundred. In theory the same range is used by the Emotiv Epoc headset, but in practice a player's emotions never reach zero and never reach above 90. In order to improve the output of the Emotion model it would be beneficial to normalize the values from the Emotiv Epoc headset such that the lowest emotion value a player feel is set to zero, and the highest to 100. This would make the two Emotion modules more compatible and would improve the implementation of the Emotion module.

The Emotion Learner could benefit from improvements in two areas:

1. More accurate emotion output from fine tuning the artificial neural network
2. Directed emotions

6.1.1.3 Emotion module

As a result of the implementation decisions made in the both the Emotion Learner and the Emotion Logger the Emotion module by Carlson and Hellevang (2010) was expanded by three new emotions. One of the key features of the EEG-based Emotion module is the ability to change the model used for emotions. In this thesis a country-specific and a general model is created, and evaluated.

6.1.2 Evaluation of Performance

To measure the effects the new Emotion module configurations have on performance simulations of the game were run using these different configurations. The evaluation results show that the Emotion module decreases performance overall in terms of supply centers.

6.1.2.1 Game state or EEG based emotions

In the simulations run by Carlson and Hellevang (2010) only Germany produced statistically significant results. In the simulations the country-specific Emotiv Epoc emotions performs better than the game state based emotions created by Carlson and Hellevang. Although there are very few significant findings, the results indicate that the EEG-based emotions can for some countries perform better.

6.1.2.2 Country specific or general emotions

When comparing the results of the country specific and general emotions there are two conflicting results. The Emotion module using both game state based and EEG-based emotions perform better with the general emotions than with the country-specific emotions. In direct contradiction to this the configuration using only the EEG-based emotions performed best using the country specific emotions. The performance difference is bigger in the EEG-based simulations, indicating that country specific emotions do perform better than general emotions.

6.2 Future works

6.2.1 Player Testing

This thesis discusses how emotion modeling can improve the agent's believability in section 2.2.3.1. Whether this is the case for the EEG-based Emotion module is something which has not been explored. A player test conducted by Carlson and Hellevang suggests that playing against emotional agents is more fun (Carlson & Hellevang, 2010). Whether this is also true for EEG-based agents could be interesting to explore further.

6.2.2 Affective wearables

Using affective wearables, such as the Emotiv Epoc headset, has been shown to have applications in research. One of these applications has been demonstrated in this thesis. In the following sections some further applications will be suggested and briefly discussed.

6.2.2.1 Emotions in decision making

The work done in this thesis opens up for the possibility to study the impact emotions has on decision making with the help of real time data about a player's emotions. Data generated from the Emotion Logger has information about the player's emotions and the order the player took while experiencing the emotions. This would make it possible to do analysis or modeling that could determine what behavior each emotion promotes.

6.2.2.2 Facial expressions

The Emotiv Epoc headset used in this thesis is also able to give data about the users facial expression. The facial expression data could be used to model emotions more accurately when the accuracy of the EEG-data about emotions is inadequate. In a social game, like StateCraft, it would also be possible to model the players' facial expressions in the same way the emotions are modeled in this thesis. This would give the possibility of displaying a facial expression for each agent to the user. By analyzing player's facial expressions the emotion model could be improved. For instance, if a player is smiling he may be feeling joy, or if the player is frowning he may be angry or frustrated.

6.2.2.3 Real time emotion modeling

With the help of affective wearables it would also be possible to model emotions in real time, which could have numerous application. In StateCraft agents would be able to identify emotions, and react to them, if the player is wearing an EEG-device. Real time emotion modeling could be very useful in environments where agents need to change behavior based on the users emotions. For instance, in a learning environment the agent needs to challenge the user appropriately, and keep them from becoming too frustrated with the given tasks.

6.3 Emotion module

The Emotion module developed in this thesis opens up for further research opportunities. By using a different EEG-device it could be possible to gather data about more emotions, which would be possible to implement into the Emotion module. As mentioned in 6.2.2.2 it would also be a possibility to expand or improve the Emotion module by using facial expressions to give more data about emotions.

This thesis uses artificial neural networks. It is possible that this is not the best technique to use, or that it is not used in the best way. Further testing and evaluation may show ways to improve both the emotion model and how it was created.

References

Azcarraga, J., Ibañez, J. F., Lim, I. R., Lumanas, N. J., Trogo, R., & Suarez, M. T. (2011). Predicting Academic Emotion based on Brainwaves Signals and Mouse Click Behavior. Chiang Mai, Thailand: Asia-Pacific Society for Computers in Education.

Bates, J. (1994). The Role of Emotion in Believable Agents. *The Role of Emotion in Believable Agents*. Communications of the ACM.

Belavkin, R. V. (2001). The Role of Emotion in Problem Solving. Helsington: AISB Press.

Calhamer, A. (1974). *The Invention of Diplomacy*. Hentet 03 08, 2011 fra The Diplomatic Pouch: http://www.diplom.org/~diparch/resources/calhamer/invention.htm

Carlson, C., & Hellevang, M. (2010). *Improving user experience in StateCraft*. Bergen: University of Bergen.

Clore, G. L., & Centerbar, D. B. (2004). Analyzing Anger: How to Make People Mad. I *Emotion* (ss. 139-144). American Psychological Association.

Copeland, J. (2000). *A Brief History of Computing*. Hentet 03 10, 2012 fra http://www.alanturing.net/turing_archive/pages/reference%20articles/briefhistofcomp.html

El-Nasr, M. S., & Skubic, M. (1999). A Fuzzy Emotional Agent for Decision-Making in a Mobile Robot. Texas A&M University Press, Texas US.

Emotiv. (2011). *Emotiv Homepage*. Hentet 03 27, 2011 fra Emotiv WebSite: http://www.emotiv.com

Emotiv. (2011). Emotiv Software Development Kit - User Manual for Release 1.0.0.4.

Fahlman, S. E. (1988). *An Empirical Study of Learning Speed in Back-Propagation Networks*. CMU.

GitHub. (2012). *Request an educational account - GitHub*. Hentet 06 05, 2011 fra https://github.com/edu

Heaton Research. (2012). *Encog Java and DotNet Neural Network Framework*. Hentet 03 15, 2012 fra http://www.heatonresearch.com/encog

Heaton, J. (2011, September). *Getting Started with Encog3 in Java*. Hentet 12 21, 2011 fra http://www.heatonresearch.com/dload/ebook/StartingEncog3Java.zip

Helgesen, A. S., & Krzywinski, A. (2006). *The Caenus Architecture - An Agent Architecture for Social Board Games*. Bergen: Department of Information Science and Media Studies, University of Bergen.

Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design Science in Information Systems Research. *MIS Quarterly Volume 28 Number 1*, 75-105.

Joone Project on SourceForge. (2012). *JOONE - an Object Oriented Neural Engine*. Hentet 03 28, 2012 fra http://sourceforge.net/projects/joone/

Kim, J., Bee, N., Wagner, J., & André, E. (2004). Emote to Win: Affective Interactions with a Computer Game Agent. University of Augsburg.

Kleiner, A. (2005). *Game AI: The shrinking gap between computer games and AI systems.* Freiburg.

Kleinginna, P. R., & Kleinginna, A. M. (1981). A Categorized List of Emotion Definitions, with Suggestions for a Consensual Definition. *Motivation and Emotion, Volume 5, Number 4*, 345-379.

Lerner, J. S., & Keltner, D. (2001). Fear, Anger, and Risk. *Journal of Personality and Social Psychology*.

Loewenstein, G., & Lerner, J. S. (2003). *The Role of Affect in Decision Making.*

McCarthy, J. (2007). *What is artificial intelligence?* Hentet February 15, 2012 fra http://www-formal.stanford.edu/jmc/whatisai/

Mitchell, T. M. (1997). *Machine Learning* (International Edition. utg.). McGRAW-HILL.

Neuroph on SourceForge. (2012). *Neuroph - Java Neural Network Framework*. Hentet 03 25, 2012 fra http://neuroph.sourceforge.net/

Ortony, A., & Turner, T. J. (1990). What's Basic About Basic Emotions? *Psychological Review, Vol 97, Number 3*, 315-331.

Ortony, A., Clore, G. L., & Collins, A. (1988). *The Cognitive Structure of Emotions.* Cambridge University Press.

Perron, B. (2005). A Cognitive Psychological Approach to Gameplay Emotions. *DiGRA 2005 Conference.* DiGRA.

Picard, R. W. (2000). *Affective Computing.* MIT Press.

Picard, R. W. (2003). Affective computing: challenges. *International Journal of Human-Computer Studies*.

Russel, S., & Norvig, P. (2003). *Artificial Intelligence: A Modern Approach* (Second Edition. utg.). Pearson Education.

Shapiro, S. S., & Wilk, M. B. (1965). *An analysis of variance test for normality (complete samples).* Biometrics.

The Code Project. (2010). *Benchmarking and Comparing Encog, Neuroph and JOONE Neural Networks*. Hentet 03 30, 2012 fra http://www.codeproject.com/Articles/85487/Benchmarking-and-Comparing-Encog-Neuroph-and-JOONE

Velásquez, J. D. (1998). *Modeling Emotion-Based Decision-Making.* Cambridge, Massachusetts: MIT Artificial Intelligence Laboratory.

Wohlin, C., Runeson, P., Höst, M., Ohlsson, M. C., Regnell, B., & Wesslén, A. (2000). *Experimentation in Software Engineering: An Introduction.* Kluwer Academic Publishers.

Woodcock, S. (2000). *Game AI: The State of the Industry*. Hentet 02 13, 2012 fra http://www.gamasutra.com/view/feature/131975/game_ai_the_state_of_the_industry.php

Appendix A

The developed code for this thesis

http://dl.dropbox.com/u/5955614/Code-ThreeProjects.zip